



UNIVERSIDAD DE LAS PALMAS DE GRAN CANARIA
Escuela de Ingeniería Informática



PROYECTO DE FIN DE CARRERA
INGENIERÍA EN INFORMÁTICA

Análisis del uso de las Redes Neuronales Artificiales en el diseño de filtros antispam

Una propuesta basada en Arquitecturas Neuronales No Supervisadas

Ylmeri Cabrera León

Tutores:

Carmen Paz Suárez Araujo

Patricio García Báez

09 de Diciembre de 2015

Carmen Paz Suárez Araujo
Catedrática de Universidad
Escuela de Ingeniería Informática
Universidad de Las Palmas de Gran Canaria

y

Patricio García Báez
Profesor Contratado-Doctor
Departamento de Ingeniería Informática y de Sistemas
Universidad de La Laguna

CERTIFICAN: que la memoria titulada "Análisis del uso de las Redes Neuronales Artificiales en el diseño de filtros antispam. Una propuesta basada en Arquitecturas Neuronales No Supervisadas" ha sido realizada por Ylermi Cabrera León bajo nuestra dirección y constituye su Proyecto de Fin de Carrera de Ingeniería en Informática.

En Las Palmas de Gran Canaria, a 09 de Diciembre de 2015



Carmen Paz Suárez Araujo

Patricio García Báez

Agradecimientos

Gracias a mi tutora, Carmen Paz, por aceptar mi propuesta no supervisada de filtro antispam. Más aún, porque siguió confiando en mí pese a su, parcial, ignorancia de los acontecimientos extraacadémicos en los que me ví envuelto y por los que me ví gravemente afectado.

Por el mismo motivo debo agradecer a mi otro tutor, Patricio, que me ayudó a perfilar y encauzar el trabajo, algunas veces desde la "lejana" Tenerife.

Agradezco a mis padres por servirme de apoyo en todos los ámbitos durante la realización de este PFC.

Termino agradeciendo a mi novia por sus ánimos, los cuales me dieron la fuerza necesaria tanto para salir del hoyo como para terminar este proyecto.

ÍNDICE

Agradecimientos.....	5
Resumen.....	11
Capítulo 1. Introducción.....	13
1.1 Motivaciones.....	14
1.2 Objetivos.....	15
1.2.1 Metodología y plan de trabajo.....	15
1.3 Correo electrónico: spam y ham.....	16
1.3.1 Formato de los mensajes.....	18
1.3.2 Proceso de envío.....	19
1.4 Computación neuronal.....	21
1.4.1 Clasificación de Redes Neuronales Artificiales.....	25
1.4.2 Mapas autoorganizados de Kohonen.....	28
1.4.3 Medición de la calidad.....	33
1.4.3.1 Calidad del mapa.....	33
1.4.3.2 Rendimiento del clasificador.....	34
1.5 Estado de la cuestión.....	36
1.5.1 Clasificación según la fuente de información.....	37
1.5.1.1 Contenido del correo electrónico.....	37
1.5.1.2 Feedback o retroalimentación del usuario.....	38
1.5.1.3 Información relativa al sistema.....	38
1.5.2 Clasificación según el método de diseño.....	39
1.5.2.1 Manuales.....	39
1.5.2.2 Basados en técnicas de aprendizaje automático.....	40
1.5.2.2.1 Supervisados.....	41
1.5.2.2.2 No supervisados.....	42
1.5.2.2.3 Semisupervisados.....	43
Capítulo 2. Diseño de filtros antispam. Cuerpo de datos y etapa de preprocesado.....	45
2.1 Cuerpo de datos.....	46
2.2 Sistema inteligente para la detección de spam.....	49
2.2.1 Etapa de preprocesado.....	51
2.2.1.1 Otros autores.....	51
2.2.1.2 Preprocesado híbrido.....	54
2.2.1.2.1 Objetivos.....	54
2.2.1.2.2 Categorías de palabras.....	54
2.2.1.2.3 Eficiencia en la implementación.....	58
2.2.1.2.4 Fases del preprocesado.....	59
2.2.1.2.5 Categorización de palabras manual vs preprocesado de la fase 3: tamaños y diferencias	65
2.2.2 Etapa de procesado.....	67
2.3 Entorno de trabajo.....	67
2.3.1 MATLAB.....	67
2.3.2 SOM Toolbox.....	68
2.3.3 Parallel Computing Toolbox.....	69
2.3.4 Hardware.....	70
Capítulo 3. Desarrollo de filtro antispam basado en mapas autoorganizados de Kohonen. Implementación y resultados.....	71
3.1 Características de la red utilizada.....	72
3.1.1 Características de la arquitectura neuronal utilizada.....	72

3.1.2 Conjuntos de datos de entrada.....	73
3.2 Descripción de funciones de los toolboxes utilizadas.....	74
3.2.1 Del SOM Toolbox.....	74
3.2.2 Del Parallel Computing Toolbox.....	74
3.3 Implementación del filtro antispam. Resultados y discusión.....	75
3.3.1 Parámetros del SOM y del preprocesado.....	75
3.3.2 Resultados y discusión.....	76
3.3.2.1 Coste temporal.....	90
Capítulo 4. Conclusiones y trabajos futuros.....	93
4.1 Conclusiones.....	94
4.2 Trabajos futuros.....	96
Bibliografía.....	99

Índice de figuras

Figura 1: Metodología evolutiva incremental.....	16
Figura 2: Estructura principal y ejemplo no real del formato de un mensaje de correo electrónico.....	18
Figura 3: Esquema explicativo del envío de un correo electrónico. Fuente: Wikipedia.....	20
Figura 4: Partes de una neurona biológica.....	21
Figura 5: Sinapsis eléctrica y química en las neuronas biológicas. Fuente: www.physicsforums.com	23
Figura 6: Partes de una neurona artificial y sus correspondientes en la neurona biológica.....	23
Figura 7: Diferentes áreas del cerebro humano forman mapas topográficos y se especializan en funciones concretas. Fuente: Wikipedia.....	28
Figura 8: Vecindad topológica con forma de función gaussiana, indicando el grado de excitación de las neuronas según la distancia lateral con la ganadora. Fuente: (Leow 2014).....	29
Figura 9: Reducción de la vecindad topológica con el paso del tiempo. Fuente: (Leow 2014).....	30
Figura 10: Diferentes tipos de vecindad: Von Neumann (4 vecinos), hexagonal (6) y Moore (8). Fuente: Gama Platform Grid Species (code.google.com).....	30
Figura 11: Los conceptos de BMU y vecindad (de Moore) sobre la estructura de un SOM (2-D). Fuente: Richard M. Clark (NTL World).....	31
Figura 12: Estabilidad de los estados: metaestable (1), estable o de mínima energía (3) e inestable (2 y 4).....	32
Figura 13: Espacio ROC, línea de decisión aleatoria y 4 clasificadores discretos: A, B, C y C'. Fuente: Wikipedia.....	36
Figura 14: División del conjunto original en tres subconjuntos: entrenamiento, ajuste y testeo.....	40
Figura 15: Esquema del sistema inteligente para la detección de spam.....	50
Figura 16: Diagrama de flujo sobre las relaciones entre las distintas fases de nuestro preprocesado.....	59
Figura 17: Diagrama de flujo de la fase 1 de nuestro preprocesado.....	61
Figura 18: Diagrama de flujo de la fase 2 de nuestro preprocesado.....	62
Figura 19: Diagrama de flujo de la fase 3 de nuestro preprocesado.....	64
Figura 20: Diferencia entre los conjuntos A y B (en color amarillo). Fuente: matematicaabelortega.blogspot.com.es	65
Figura 21: Curvas ROC de 8 mejores SOM Top 25 – Escenario 1 – Fase de Entrenamiento-Ajuste.....	86
Figura 22: Curvas ROC de 5 mejores SOM Top 10 – Escenario 1 – Fase de Entrenamiento-Ajuste.....	86
Figura 23: Curvas ROC de 5 mejores SOM Top 10 – Escenario 2 – Fase de Entrenamiento-Ajuste.....	87
Figura 24: Curvas ROC de 2 mejores SOM Top 25 – Escenario 2 – Fase de Entrenamiento-Ajuste.....	87
Figura 25: Curva ROC del mejor SOM Top 50 – Escenario 2 – Fase de Entrenamiento-Ajuste.....	88
Figura 26: Curva ROC del mejor SOM Top 75 – Escenario 2 – Fase de Entrenamiento-Ajuste.....	88
Figura 27: Curvas ROC de 2 mejores SOM Top 95 – Escenario 2 – Fase de Entrenamiento-Ajuste.....	89
Figura 28: Curvas ROC de 2 mejores SOM Top 100 – Escenario 2 – Fase de Entrenamiento-Ajuste.....	89
Figura 29: Curvas ROC de los 2 mejores SOM – Ambos escenarios – Fase de Testeo.....	90

Índice de tablas

Tabla 1: Características de las RNB y de las RNA.....	24
Tabla 2: Características del computador de Von Neumann original y de las RNB.....	25
.....Tabla 3: Matriz de confusión o tabla de contingencia: TP, FP, TN y FN.	35
Tabla 4: Repositorios de correos electrónicos públicos, disponibles libre y gratuitamente.....	49
Tabla 5: Conjunto de datos seleccionado. Se han destacado en negrita los dos subconjuntos de correos electrónicos que conformarán el de nuestra investigación, nuestro cuerpo de datos.....	49
Tabla 6: Métodos de selección de características utilizados por otros investigadores.....	53
Tabla 7: Dimensiones de los vectores de entrada a la red utilizadas en otros trabajos.....	54
Tabla 8: Clasificación visual tipo semáforo de las 13 categorías de palabras según su grado de spaminess..	55
Tabla 9: Categorías de spam existentes según diversos autores (destacado en negrita el nuestro).....	57
Tabla 10: Porcentajes de ham y spam de las categorías 1-6 en todo nuestro cuerpo.....	58
Tabla 11: Porcentajes de ham y spam de las categorías 7-13 en todo nuestro cuerpo.....	58
Tabla 12: Porcentajes de ham y spam de las categorías 1-6 en el conjunto entrenamiento-ajuste.....	58
Tabla 13: Porcentajes de ham y spam de las categorías 7-13 en el conjunto entrenamiento-ajuste.....	58
Tabla 14: Cantidades de palabras de las categorías 1-6 según procedimiento aplicado.....	66
Tabla 15: Cantidades de palabras de las categorías 7-13 según procedimiento aplicado.....	66
Tabla 16: Cantidades de palabras de las categorías 1-6 en la diferencia de los procedimientos.....	66
Tabla 17: Cantidades de palabras de las categorías 7-13 en la diferencia de los procedimientos.....	66
Tabla 18: Palabras incluidas en las diferencias de conjuntos de algunas de las categorías, las de menor tamaño.....	66
Tabla 19: Conjunto de soluciones finales: ranking de clasificadores según AUC (ambos escenarios).....	78
Tabla 20: Rangos o conjuntos de valores de MQE y TE obtenidos por otros autores.....	79
Tabla 21: Recuento y calificación de clasificadores en cada rango de AUC, para ambos escenarios.....	79
Tabla 22: Métricas obtenidas para los 1-6 mejores clasificadores del escenario 1 (Fase de Entrenamiento-Ajuste).....	81
Tabla 23: Métricas obtenidas para los 7-13 mejores clasificadores del escenario 1 (Fase de Entrenamiento-Ajuste).....	82
Tabla 24: Métricas obtenidas para los 1-6 mejores clasificadores del escenario 2 (Fase de Entrenamiento-Ajuste).....	83
Tabla 25: Métricas obtenidas para los 7-13 mejores clasificadores del escenario 2 (Fase de Entrenamiento-Ajuste).....	84
Tabla 26: Métricas obtenidas para los mejores clasificadores de ambos escenarios (Fase de Testeo).....	85
Tabla 27: Coste temporal y funcionalidades de algunas de las funciones implementadas.....	91

Resumen

En la actual Era de la Información, el uso e importancia de las comunicaciones telemáticas ha crecido, primordialmente debido al auge de las TIC. Dentro de las múltiples formas de realizar dicha comunicación destaca el correo electrónico o email, principalmente por ser bastante utilizado desde hace décadas. Lamentablemente, esta popularidad ha venido acompañada de amenazas como bulos, ataques cibernéticos, virus informáticos y, en mayor medida, spam.

Se considera spam todo aquel mensaje de correo electrónico enviado masivamente sin que los receptores lo hayan deseado ni solicitado. Resaltamos lo de "masivo" puesto que, desde hace años, presenta un alto volumen de circulación: unos 62 billones anuales, es decir, solamente 3 de cada 10 mensajes pueden considerarse como sí deseados (ham). Como además, en la mayoría de casos, su contenido es ofensivo o dudoso (*e.g.* estafas, *phishing*, medicamentos ilegales, pornografía, réplicas...), podemos afirmar que es una gran lacra, generando pérdidas temporales y económicas cuantiosas.

Este Proyecto de Fin de Carrera presenta una solución a este problema, así como un análisis del uso de las redes neuronales artificiales (RNA) para ello. Se ha propuesto un sistema inteligente híbrido de filtrado antispam, basado en RNA no supervisadas. Dicho sistema consta de una etapa de preprocesado y otra de procesado, basadas ambas en distintos modelos de computación. Para el preprocesado se ha desarrollado un esquema diferente a las propuestas dadas por otros autores, constando de una fase manual y otra computacional, sustentada en computación programada. La etapa de procesado se basa en mapas autoorganizados de Kohonen (*SOM*). Este sistema ha sido optimizado usando, como cuerpo de datos, ham de "Enron Email" y spam de dos fuentes diferentes. Se ha analizado la calidad y el rendimiento del mismo mediante las medidas de calidad del mapa de Kohonen más utilizadas, *MQE* y *TE*, y todo el conjunto de métricas de rendimiento de un clasificador, al uso.

Esta memoria describe los procesos de análisis, diseño, implementación y verificación del mencionado filtro antispam y está estructurada en 4 capítulos, con el siguiente contenido:

- Capítulo 1: se establece el ámbito de desarrollo del proyecto (correos electrónicos y Redes Neuronales Artificiales), indicando los motivos de la realización del mismo, los objetivos perseguidos y los antecedentes que lo preceden.
- Capítulo 2: se presentan algunos cuerpos o conjuntos de correos electrónicos libres y gratuitos, concretándose el elegido. Luego, se detallan los métodos utilizados para el tratamiento de los datos de entrada, es decir, la fase de preprocesado de estos correos electrónicos. Por último, se indican el hardware y las herramientas software con los que hemos realizado este preprocesado.
- Capítulo 3: se explica el desarrollo e implementación del filtro antispam basado en los mapas autoorganizados de Kohonen, concretamente la implementación de la fase de procesado del sistema inteligente propuesto. Se presentarán las diferentes configuraciones de *SOM* utilizadas durante el desarrollo de dicho sistema, los resultados obtenidos y la discusión de los mismos, hasta alcanzar el filtro antispam definitivo.
- Capítulo 4: se describen las conclusiones del proyecto llevado a cabo y algunos de los posibles trabajos futuros por donde se podrá continuar.

Capítulo 1. Introducción

Introducción: En este capítulo se establece el ámbito de desarrollo del proyecto, indicando los motivos de la realización del mismo y los objetivos perseguidos. A continuación, se describe el marco referencial de la investigación, es decir, se indican tanto su marco conceptual (*i.e.* especificación de términos e ideas relevantes con el fin de evitar confusiones o malinterpretaciones) como su marco teórico (*i.e.* antecedentes e investigaciones recientes a través de la revisión de la literatura relacionada).

1.1 Motivaciones

En la actual Era de la Información, la importancia de las comunicaciones telemáticas se ve cada día incrementada debido a la preponderancia de sistemas de información y redes de ordenadores, tanto dentro como fuera de organizaciones empresariales y académicas. Una manera de llevar a cabo esta comunicación y que cuenta ya con varias décadas de funcionamiento es mediante el correo electrónico¹, siendo sus principales características su rapidez y gratuidad². Esto ha dado lugar a un gran auge en su uso y difusión. Desafortunadamente, ello ha venido acompañado por una serie de amenazas tales como, entre otras, virus informáticos, bulos (*hoax*), ataques cibernéticos y, sobre todo, spam³. Sobre esta última nos centraremos durante el presente proyecto.

El spam surgió de la utilización inadecuada e indeseada del correo electrónico como medio a través del cual realizar el envío masivo de correos no deseados ni tampoco solicitados (en contraposición al ham o correo legítimo y sí deseado) por parte de los *spammers*, siendo estos, en la mayoría de los casos, empresas que desean ofertar algún producto o servicio, tanto propio como de terceros (Stringhini et al. 2014).

Lo que comenzó como un mero incordio para los pocos usuarios de correo electrónico que existían al principio evolucionó hacia un problema con cierta relevancia por dos motivos, primordialmente:

- Alto volumen de circulación: alrededor del 71% de todos los correos electrónicos pueden ser considerados como no deseados (Trustwave 2013a), unos 62 billones durante 2008 (McAfee y ICF International 2009).
- Contenido ofensivo o dudoso: pornografía, medicamentos ilegales, adivinación, *phishing*⁴, réplicas, estafas... (Trustwave 2013b)

Todo ello ha afectado tanto a empresas como a meros usuarios al haber generado graves perjuicios tanto de tiempo como económicos: pérdidas de 20 mil millones de dólares anuales frente a un beneficio de los *spammers* por valor de 200 millones de dólares (Rao y Reiley 2012; Wiehes 2005; Lieb 2002). Es por ello que muchos países o regiones se han visto obligados a adaptar o adoptar algunas medidas legislativas al respecto (Cabrera y Acosta 2011):

- la LOPD⁵ y la LSSI⁶, en España.

1 Suele ser bastante frecuente la utilización del término original en inglés, “*email*”, contracción de la palabra “*electronic mail*”. El uso de dicho vocablo comenzó a ser común *circa* 1993 (Wikipedia 2014b).

2 Resulta evidente *a posteriori* que enviar un correo electrónico no es gratuito debido al coste temporal (tiempo de cómputo, tiempo perdido por el usuario receptor en su clasificación manual...) y al de otros recursos (electricidad, emisiones de gases de efecto invernadero, almacenamiento, procesamiento, etc). Según (McAfee y ICF International 2009), el coste de enviar un spam es de 0,3 gramos de CO₂, equivalente a conducir 1 metro o, multiplicado por el volumen anual de spam, dar la vuelta a la Tierra 1,6 millones de veces.

3 No confundir la palabra “spam” con “Spam” o “SPAM” (posiblemente acrónimo de “*SPiced hAM*” o “*Shoulders of Pork And hAM*”, “jamón condimentado/especiado” o “paleta de cerdo y jamón”, respectivamente), un producto cárnico enlatado similar al jamón, elaborado por la empresa Hormel Foods Corporation desde 1937.

4 Suplantación de identidad con el fin de adquirir información sensible o confidencial (nombres de usuario, contraseñas, tarjetas de crédito...) de forma fraudulenta a través de una comunicación electrónica (Ej.: correo electrónico, teléfono, mensajería instantánea, redes sociales, etc).

5 Ley Orgánica 15/1999, de 13 de Diciembre, de Protección de Datos de Carácter Personal (Cortes Generales de España 1999).

6 Ley 34/2002, de 11 de Julio, de Servicios de la Sociedad de Información y Comercio Electrónico (Cortes Generales

- la Directiva 2002/58/EC sobre Privacidad y Comunicaciones Electrónicas⁷, en la UE.
- la *CAN-SPAM⁸ Act of 2003*, en EEUU.

Los filtros antispam nacieron con la idea de detener el aumento en la proliferación de correos no deseados que comenzó a aflorar allá por 1994 (Lutus 2006), métodos que en la mayoría de los casos eran manuales. Dichos primeros sistemas de filtrado clasificaban un mensaje de correo electrónico como spam o ham basándose tanto en la dirección de correo electrónico y país de origen del spam como en el contenido del mismo, por lo que no eran ni rápidos ni completamente efectivos. Con el devenir de los años, y como suele ocurrir con todos los sistemas de seguridad y de defensa en diferentes ámbitos, se ha confirmado que los atacantes (*spammers* en nuestro caso) van siempre un paso por delante (Ramachandran y Feamster 2006; Postini 2004; Wang, Irani y Pu 2013) de ahí que haga falta estar continuamente haciendo frente a las nuevas amenazas y contrarrestando las carencias, debilidades y fallos de seguridad que los *spammers* encuentran en dichos filtros (u otro software o hardware), y de los cuales se aprovechan (Posluns y Sjouwerman with Spammer-X 2004).

Teniendo en consideración todo lo expuesto, el tema del filtrado de correos no deseados tiene una importancia considerable, siendo bastante extenso tanto el número de investigaciones y trabajos realizados sobre ello como la cantidad de aplicaciones y métodos seguidos para diseñar y crear sistemas antispam. Por ello, en este proyecto nos centraremos en el filtrado antispam o, dicho de otro modo, en la detección de correos spam.

1.2 Objetivos

Este Proyecto de Fin de Carrera aborda un problema complejo y de alta incidencia socioeconómica debido al tipo de sociedad en la que vivimos, Sociedad de la Información y el Conocimiento: la alta proliferación de spam que circula por Internet.

Los objetivos propuestos pretenden dar un paso hacia adelante en su conocimiento, en la forma de abordarlo y en la mejora de su solución. Así, los objetivos fundamentales del mismo son:

- Análisis de filtros antispam basados en Redes Neuronales Artificiales (RNA).
- Desarrollo de un método de preprocesado del cuerpo de datos reales para la obtención de un entorno de información que optimice los filtros antispam a diseñar.
- Diseño, desarrollo, implementación y análisis de un sistema de filtrado antispam basado en RNA con aprendizaje no supervisado.

1.2.1 Metodología y plan de trabajo

Para alcanzar los objetivos propuestos se ha seguido una metodología evolutiva incremental (Figura 1) con la que es posible la obtención de versiones parciales previas a la final, cada una de las cuales incorpora nuevas funcionalidades y requisitos que fueron analizados como necesarios al

de España 2002).

7 Directiva 2002/58/CE del Parlamento Europeo y del Consejo, de 12 de julio de 2002, relativa al tratamiento de los datos personales y a la protección de la intimidad en el sector de las comunicaciones electrónicas (Parlamento Europeo y Consejo 2002).

8 *Controlling the Assault of Non-Solicited Pornography And Marketing* o Controlar el Asalto de Pornografía y Marketing No Solicitado (United States Senate y United States House of Representatives 2003).

principio.

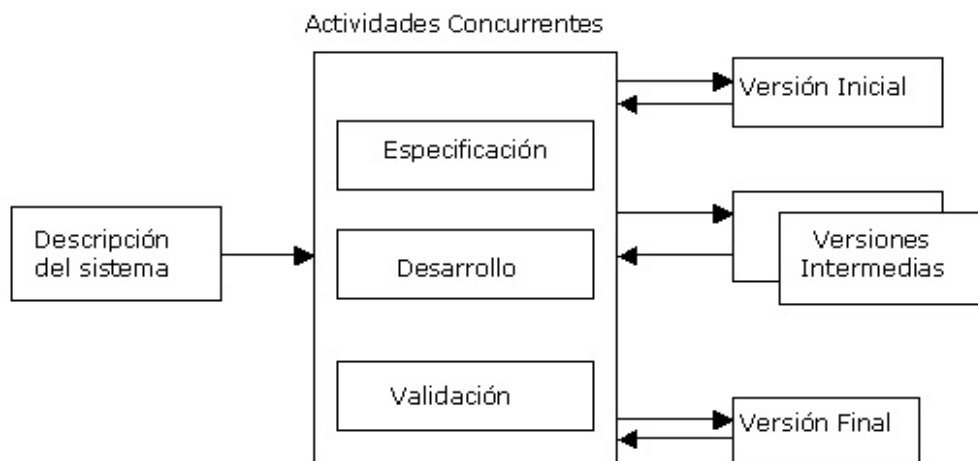


Figura 1: Metodología evolutiva incremental.

El plan de trabajo seguido se compone del siguiente orden de tareas:

1. Estudiar el tema de los correos electrónicos: terminología, tecnología, filtros, etc.
2. Estudiar las Redes Neuronales Artificiales, poniendo mayor énfasis en las Arquitecturas Neuronales No Supervisadas.
3. Analizar las investigaciones precedentes relativas al filtrado antispam, tanto si se basan como si no en RNA.
4. Realizar un análisis de los diversos *corpora* de correos electrónicos existentes.
5. Adquirir soltura en el uso de las herramientas informáticas (Ej.: entorno de desarrollo, *toolboxes*⁹, lenguaje de programación, Sistema Operativo...) que se pretendan utilizar.
6. Desarrollar un método de preprocesado de los datos.
7. Construir y analizar un filtro antispam basado en RNA con aprendizaje no supervisado.

1.3 Correo electrónico: spam y ham

El correo electrónico es un servicio de red por el cual se intercambian mensajes digitales entre un emisor y uno o varios receptores a través de Internet u otras redes de ordenadores.

Como es habitual en las telecomunicaciones, cada estándar es definido mediante la divulgación de su correspondiente *Request For Comments (RFC)* o Petición de Comentarios, un conjunto, en la actualidad bastante numeroso, de publicaciones con la descripción de, principalmente, protocolos, procedimientos, comentarios e ideas sobre diversos aspectos del funcionamiento de Internet y otras redes. Las *RFC* son publicadas por la *Internet Engineering Task Force (IETF)*, consorcio de colaboración técnica más importante en Internet) como unos documentos en formato *ASCII*, en lengua inglesa y manteniendo entre ellas una cierta estructura. Son identificadas unívocamente mediante un número: 706 (sobre el problema del correo no deseado), 768 (protocolo de transporte UDP), 3629 (conjunto de caracteres UTF-8), 4250 (administración remota SSH), 4880 (criptografía de clave pública PGP), etc. Antes de que un documento sea

⁹ Conjunto de comandos u operaciones que se puede agregar opcionalmente a un programa informático (principalmente, a un Sistema Operativo o a un entorno de desarrollo) con el fin de aumentar su funcionalidad.

considerado como *RFC* debe seguir un proceso muy estricto para asegurar su calidad y coherencia tras haber superado críticas, objeciones y sugerencias, por lo que el nombre se ha quedado obsoleto.

La datación del correo electrónico es bastante difusa ya que, dependiendo del motivo aducido, podemos elegir las siguientes fechas (Wikipedia 2014b):

- 1962: año en el que se instaura un sistema de mensajería electrónica sobre la antigua red digital militar *Automatic Digital Network (AUTODIN)*.
- 1971: el primer correo electrónico es mandado sobre la *Advanced Research Projects Agency Network (ARPANET)*, red militar considerada como la precursora de Internet, ambas basadas en la familia de protocolos de Internet TCP/IP, formada por la unión de los protocolos TCP (*Transmission Control Protocol* o Protocolo de Control de la Transmisión; protocolo de la capa de transporte, estructurado en segmentos TCP, con control de flujo, control de congestión, y detección de errores; especificado por primera vez en la *RFC 675*) e IP (*Internet Protocol* o Protocolo de Internet; protocolo de la capa de red encargado de enrutar los paquetes o datagramas IP entre dos máquinas; explicado desde la *RFC 791*).
- 1973: se propone un protocolo de correo basado en el protocolo FTP¹⁰ (*RFC 524*) y, varios meses después, el primer intento de estandarización u homogeneización de las cabeceras de los mensajes de red (*RFC 561*).
- 1977: los mensajes sobre *ARPANET* son estandarizados por la *RFC 733*.
- 1982: la *RFC 822* estandariza los mensajes sobre la red Internet. Posteriormente actualizada en 2001 por la *RFC 2822*, en 2008 por la *RFC 5322*, y en 2013 por la *RFC 6854*.

Como ya hemos visto, este servicio es considerado como un veterano dentro de los sistemas de comunicación electrónica aunque aún no se ha quedado desfasado porque se ha ido modernizando y extendiendo, manteniendo todavía compatibilidad hacia atrás, con la introducción de ciertas características. Éstas vienen estipuladas en las *MIME (Multipurpose Internet Mail Extensions* o Extensiones Multipropósito de Correo de Internet, descritas extensamente en las *RFC 2045, 2046, 2047, 4288, 4289 y 2049*), las cuales permiten que *SMTP* soporte:

- Texto en diversos formatos de codificación de caracteres internacionales: originalmente sólo se admitían caracteres *ASCII*¹¹.
- Información de encabezados con conjuntos de caracteres diferentes de *ASCII*.
- Posibilidad de incorporar ficheros adjuntos multimedia (texto, imagen, audio, vídeo, ejecutable¹², etc): en sus comienzos únicamente se admitía texto pero no como adjunto sino dentro del propio cuerpo del mensaje.
- Cuerpos de mensajes con múltiples partes (*multi-part*).

Relativo a su clasificación, cualquier mensaje de correo electrónico puede ser identificado, *a priori* fácilmente, como spam o ham. Lamentablemente y como veremos en el apartado 1.5, esto es un proceso parcialmente subjetivo puesto que lo que para una persona es un ham, para otra podría

10 *File Transfer Protocol* o Protocolo de Transferencia de Ficheros. Se trata de un protocolo de red estándar para, como su propio nombre indica, la transferencia de archivos entre dos dispositivos sobre una red basada en el protocolo TCP, como es el caso de *ARPANET* o Internet. Viene explicado en la *RFC 959* y hace uso del puerto 21.

11 *American Standard Code for Information Interchange* o Código Estándar Estadounidense para el Intercambio de Información. Se trata de un sistema de codificación de caracteres estándar por el cual se utilizan, por carácter, sólo 7 bits (8 si tenemos en cuenta el bit de paridad para control de errores), de ahí que sea bastante limitado en comparación con otros formatos como el estándar Unicode (más de 113.000 caracteres diferentes en la versión 7.0) ya que únicamente es capaz de distinguir entre 128 caracteres, tanto imprimibles (letras mayúsculas y minúsculas del inglés moderno, números, y símbolos de puntuación) como de control (escape, retorno de carro, eliminar, etc).

12 Muchos gestores de correo electrónico impiden la transmisión de mensajes con ficheros adjuntos ejecutables, estén o no encapsulados en ficheros comprimidos, debido a que traen aparejado riesgo de infección por *malware* como virus, troyanos, gusanos, espías, *keyloggers* (principalmente, para robar contraseñas y números de tarjetas de crédito), etc.

ser un spam, o viceversa (Graham-Cumming 2006), de ahí que hayan aparecido términos para señalar esta dificultad (*easy ham* y *hard ham* (Spam-Assassin 2013)) o ambigüedad ("bacn"¹³ y "casos grises"¹⁴). Por ello, es muy difícil conseguir un filtro que sea idóneo para todos los usuarios.

En los siguientes subapartados explicaremos tanto el formato de los mensajes de correo electrónico como el proceso de envío de los mismos, definiéndose los métodos antispam en base a ambos, principalmente.

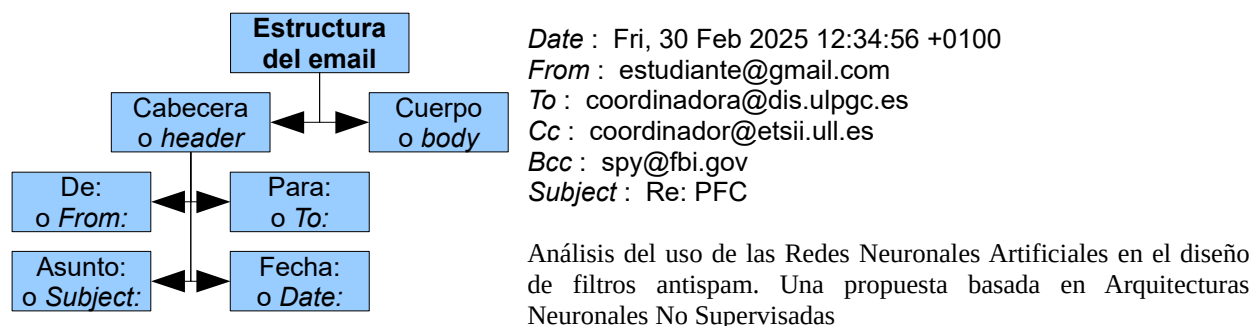


Figura 2: Estructura principal y ejemplo no real del formato de un mensaje de correo electrónico.

1.3.1 Formato de los mensajes

El formato de los mensajes (más conocido por las siglas *IMF*¹⁵) viene definido en la *RFC* 5322 (P. Resnick 2008), y tal y como se puede apreciar en la Figura 2, cada mensaje se puede estructurar en tres partes de las que únicamente nos interesarán las dos primeras ya que son las únicas que se pueden utilizar, debido al carácter temporal de la tercera, para la clasificación de correos electrónicos en destino, es decir, en el receptor del mensaje:

- **Cabecera (*header*):** estructurada en distintos campos tales como, entre otros muchos, "*From*:", "*To*:", "*CC*:", "*BCC*:", "*Subject*:", y "*Date*:" cuyo significado es, respectivamente, supuesta¹⁶ dirección del emisor, supuestas direcciones de los receptores, direcciones para la Copia de Carbón, direcciones para la Copia de Carbón Oculta (similar al anterior pero los destinatarios reciben el mensaje sin aparecer en ninguna lista), línea de asunto, fecha y hora de envío (normalmente definida automáticamente por el cliente de correo).
- **Cuerpo (*body*):** contenido básico, como texto sin formato, con bloque de firmas final opcional. Sin límite de tamaño. Está separado de la cabecera por una línea en blanco.
- **Sobre (*envelope*):** agregado automáticamente al enviar el mensaje y eliminado nada más recibirlo, de ahí su carácter caduco. Es utilizado por el protocolo *SMTP*, que explicaremos en el apartado 1.3.2, para la transferencia del propio mensaje. Contiene toda la información

13 Mensajes solicitados de carácter comercial pero que casi nunca o nunca son leídos por el receptor por recibirse en grandes cantidades o en momentos inoportunos (Wikipedia 2013a).

14 *Grey cases*. Son mensajes que no quieres, tal vez no todos los recibidos, pero que quizá fueron solicitados en su momento (James Bruce 2012). Ejemplos: notificaciones diarias de Groupon, actualizaciones de Facebook, boletines informativos, listas de correos, etc.

15 *Internet Message Format* o Formato de Mensaje de Internet.

16 Las direcciones de los receptores a los que realmente se hace el envío vienen indicadas por el protocolo *SMTP* en el sobre del mensaje y pueden ser completamente diferentes a las del campo "*To*:". Lo mismo ocurre con la del emisor en el campo "*From*:".

necesaria para cumplir con la transmisión y entrega del mensaje (Klensin 2008): dirección de correo del creador del mensaje (a quién se debe dirigir en caso de error), dirección de al menos un receptor, y parámetros de envío opcionales (Chhabra 2005).

1.3.2 Proceso de envío

El envío de mensajes de correo electrónico (Figura 3) tiene un funcionamiento similar al de su homólogo, el correo postal, entendiéndose las oficinas o estafetas de Correos como símil de servidores de correo (*Mail Transfer Agent*, *MTA*), y los carteros o mensajeros como símil de *Mail Delivery Agent* (*MDA*).

En la actualidad, este intercambio de mensajes de correo electrónico entre dos dispositivos se fundamenta en el protocolo de red *Simple Mail Transfer Protocol* (*SMTP*) o Protocolo para la Transferencia Simple de Correo Electrónico, proceso que se realiza a través del puerto 25. Es considerado estándar oficial de Internet al venir definido desde 1982 en la *RFC* 821, y renovado en la *RFC* 5321 (Klensin 2008). Adicionalmente y debido a la simpleza de *SMTP*, para la recepción de los mensajes frecuentemente se hace uso de alguno de los siguientes protocolos de aplicación:

- *POP3: Post Office Protocol* o Protocolo de Oficina Postal, en su versión 3 (*RFC* 1939). Permite obtener los mensajes de correo electrónico almacenados en un servidor de Internet remoto a través de conexiones TCP/IP. Aventura al *IMAP* en que el número de transacciones con el servidor de correo es mucho más reducido (normalmente una inicial para descargar todos los correos, lamentablemente se vayan o no a leer, más las sucesivas cada cierto número de minutos en vez de la conexión permanente de *IMAP*), o sea, permite modo de trabajo *offline*.
- *IMAP4rev1: Internet Message Access Protocol* o Protocolo de Acceso a Mensajes de Internet, en su versión 4 revisión 1 (*RFC* 3501). Tiene la misma funcionalidad que *POP3* aunque es más complejo debido a que es capaz de: etiquetar mensajes, hacer uso de carpetas, mantener copias de los mensajes en el servidor (en vez de descargarlos y borrarlos de éste, opción por defecto en *POP3*), permitir conexiones simultáneas a la misma cuenta de correo, permitir aviso de llegada de correo nuevo inmediatamente, sólo descargar ciertas partes de los mensajes (por ejemplo, ignorar los ficheros adjuntos), buscar mensajes con ciertas características o palabras clave, etc.

Seguidamente explicaremos el esquema de envío típico en el cual tanto el emisor, o remitente, como el receptor hacen uso de un cliente de correo local (*Mail User Agent*, *MUA*):

1. Ana (emisor) quiere enviar un mensaje a Bea (receptor). Tras redactarlo, lo envía desde su *MUA* al servidor de correo remoto (normalmente un *MTA*) que tiene asociado, a través del puerto 25, mediante el protocolo *SMTP*.
2. Este servidor de correo sabe, debido a que viene indicada en el sobre *SMTP* del mensaje, que la dirección del receptor pertenece al dominio “b.com” de forma que le pregunta al servidor *DNS* (*Domain Name System* o Sistema de Nombres de Dominio) de dicho dominio (“ns.b.com”) por su registro *MX* (*Mail eXchange record* o registro de intercambio de correo), que especifica cómo debe ser encaminado un correo electrónico en Internet al indicar un servidor de correo responsable de aceptar mensajes pertenecientes al dominio del receptor (“b.com”). Estos registros *MX* apuntan, por un lado, a los servidores a los cuales envían un correo electrónico, y, por otro lado, a cuál de ellos debería ser enviado en primer lugar, ya sea por prioridad o por balanceo de carga.
3. Dicho servidor *DNS* le responde con los registros *MX*, si los supiera, que listan los servidores de intercambio de correo asociados al dominio deseado (“mx.b.com”,

- denominado *MDA* en caso de que sea el servidor de correo final; *MTA* en caso contrario).
- El servidor de correo *SMTP* de Ana ya puede mandarle al *MDA* de Bea el mensaje, que es guardado.
 - Más adelante, Bea hace una petición de nuevo correo desde su *MUA* a su *MDA*, mediante el protocolo *POP3* (“pop3.b.com”; siendo el mismo ordenador que “mx.b.com”) o *IMAP*.

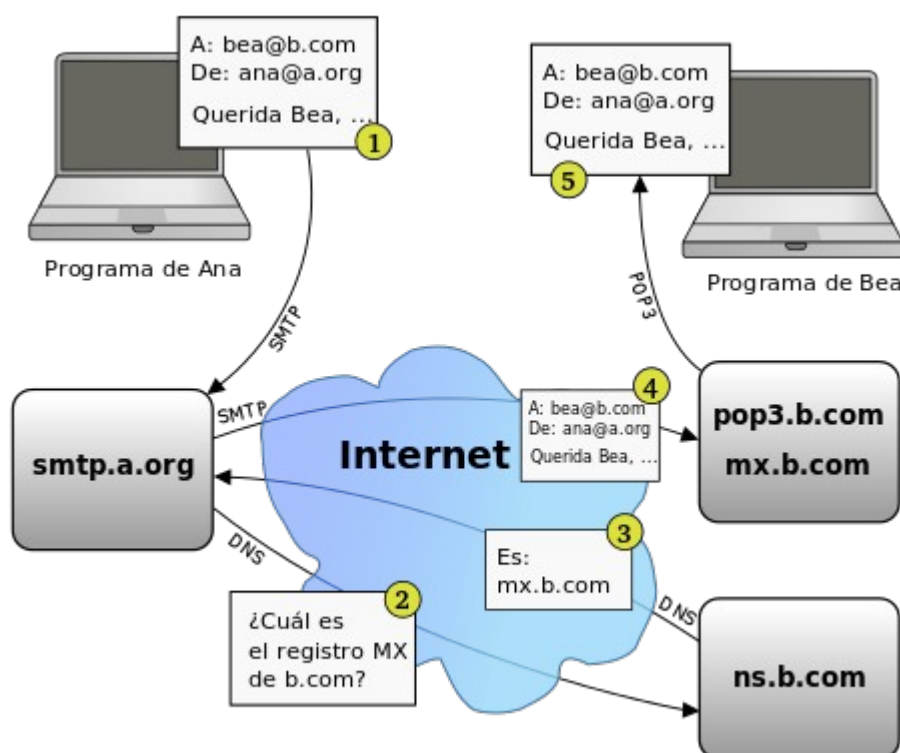


Figura 3: Esquema explicativo del envío de un correo electrónico. Fuente: Wikipedia

El proceso de envío explicado anteriormente es algo diferente en caso de utilizar algún *webmail* – *e.g.* Gmail, Outlook.com (antiguo Hotmail), Yahoo! Mail, etc – ya que en los pasos 1 y 5 la comunicación con el servidor de correo remoto se hace desde el navegador web, a través del puerto 80, mediante el protocolo HTTP¹⁷.

¹⁷ *Hypertext Transfer Protocol* o Protocolo de Transferencia de Hipertexto. Estandarizado por última vez en la RFC 7235 de Junio de 2014, es el utilizado para las transacciones de la red informática mundial *World Wide Web* (WWW), usando la red Internet como medio de transmisión.

1.4 Computación neuronal

La computación neuronal, también denominada Redes Neuronales Artificiales (RNA o ANN¹⁸), es una disciplina inspirada en las Redes Neuronales Biológicas (RNB) del Sistema Nervioso Central (SNC)¹⁹ de los animales y, más en particular, en el cerebro humano (Kohonen 1988; Basogain Olabe 1998; Gabrielsson y Gabrielsson 2006). Principalmente, los investigadores de diversos campos (ingeniería, filosofía, fisiología y psicología, entre otros) se han centrado en el análisis tanto de la composición como del funcionamiento de las RNB, de la unidad fundamental que las conforman (la neurona biológica²⁰; Figura 4) y de la interconexión de dichas neuronas (sinapsis; Figura 5). Cabe destacar la labor investigadora del científico español Santiago Ramón y Cajal a finales del siglo XIX que, por su "Doctrina de la neurona", fue premiado con el Premio Nobel de Fisiología y Medicina de 1906, siendo considerado como el padre de la neurociencia moderna (Santiago Ramón y Cajal 1917).

(Kohonen 1988) define las RNA como *"redes, interconectadas masivamente en paralelo, de elementos simples (normalmente adaptativos) organizados jerárquicamente, las cuales intentan interactuar con los objetos del mundo real del mismo modo que lo hace el sistema nervioso biológico"*. Presentan las siguientes características propias del cerebro (Jain, Mao y Mohiuddin 1996; Basogain Olabe 1998):

- Aprender: adquirir conocimiento en base a la experiencia y a la práctica. Las RNA pueden moldear su comportamiento en función del entorno, aprendiendo mediante la variación adaptativa de los pesos sinápticos²¹ (Rumelhart, Hinton y McClelland 1986).
- Generalizar: abstraer lo que es común y esencial a muchas cosas, para formar un concepto general que las comprenda todas. Las RNA son parcialmente tolerantes a fallos (debido a la existencia de redundancia), y a ruidos, distorsiones e incompletitudes en las entradas.
- Abstraer: separar por medio de una operación intelectual las cualidades de un objeto para considerarlas aisladamente. Ciertas RNA son capaces de extraer información implícita en los datos.

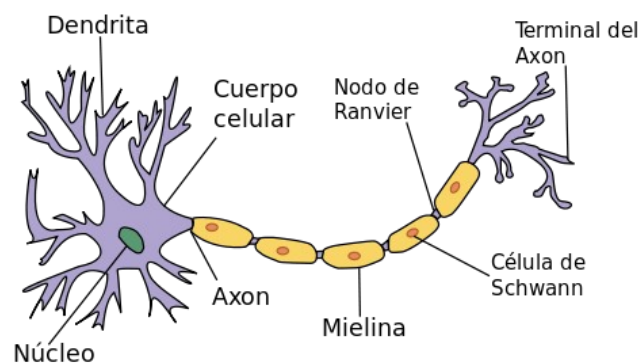


Figura 4: Partes de una neurona biológica.

18 Siglas de la palabra en inglés *Artificial Neural Network*.

19 También existe el Sistema Nervioso Periférico (SNP), el cual está formado por nervios y neuronas que residen o se extienden fuera del SNC, conectando éste con los miembros y órganos.

20 Aparte de las neuronas biológicas, en el SNC y SNP también existen las células gliales, mucho más numerosas (a razón de 50:1) y que a su vez se clasifican en varios subgrupos. Las células gliales son las encargadas de dar soporte (nutrición, defensa, sostén y aislamiento) a las neuronas además de intervenir activamente en el procesamiento cerebral al controlar tanto la composición iónica como los niveles de neurotransmisores.

21 Funcionamiento homólogo al presente en las RNB, en las que el conocimiento reside en las sinapsis.

Generalmente, las neuronas artificiales que conforman las RNA se representan esquemáticamente como en la Figura 6, en la cual también se relaciona cada una de sus partes con la correspondiente, en cuanto a la función desempeñada, en las RNB. Una neurona biológica, de las que existen muy diversos tipos y métodos de clasificación, está dividida en tres diferentes partes, principalmente (Figura 4):

- Axón: prolongación por la que transitan los impulsos nerviosos o potenciales de acción desde el cuerpo celular hacia la siguiente célula. En las neuronas de las RNA se corresponde funcionalmente con la salida.
- Cuerpo celular: también denominado pericarion o soma. Contiene al núcleo celular el cual está rodeado por el citoplasma de la neurona. Sintetiza neurotransmisores y otro tipo de moléculas. Actúa como el umbral y el sumatorio de las neuronas artificiales.
- Dendritas: prolongaciones que transmiten los impulsos recibidos desde las neuronas adyacentes hacia el soma. En cantidad variable y estructuradas heterogéneamente según el tipo de neurona. Podríamos considerarlas como las entradas en las neuronas artificiales.

La corteza o córtex cerebral es la parte más externa del cerebro humano y, además, es considerada la parte más interesante de este órgano dado que es donde se sitúan las "habilidades superiores" como la memoria, la atención, el lenguaje, el pensamiento y la conciencia. En los humanos es una capa larga y plana, de aproximadamente entre 2 y 3 milímetros de espesor, de materia gris²². Está fuertemente circonvolucionada por lo que, si se extendiese, ocuparía unos 2200 cm², cerca del doble del área que ocupa un teclado de ordenador (Jain, Mao y Mohiuddin 1996).

Resulta interesante, por un lado, analizar las características de ambas redes neuronales (Jain, Mao y Mohiuddin 1996; Gabrielsson y Gabrielsson 2006; Cabrera y Acosta 2011; Eluyode y Akomolafe 2013) (Tabla 1), y, por otro lado, comparar las propiedades de las RNB con las del computador de Von Neumann original, en el que los ordenadores contemporáneos están basados (Jain, Mao y Mohiuddin 1996) (Tabla 2).

22 La materia gris está formada por neuronas biológicas cuyos axones no están recubiertos de mielina, al contrario que en la materia blanca. La mielina tiene la función de aislar los impulsos que recorren el axón (parecido a lo que acaece en los cables blindados o apantallados), aumentando su velocidad de transmisión.

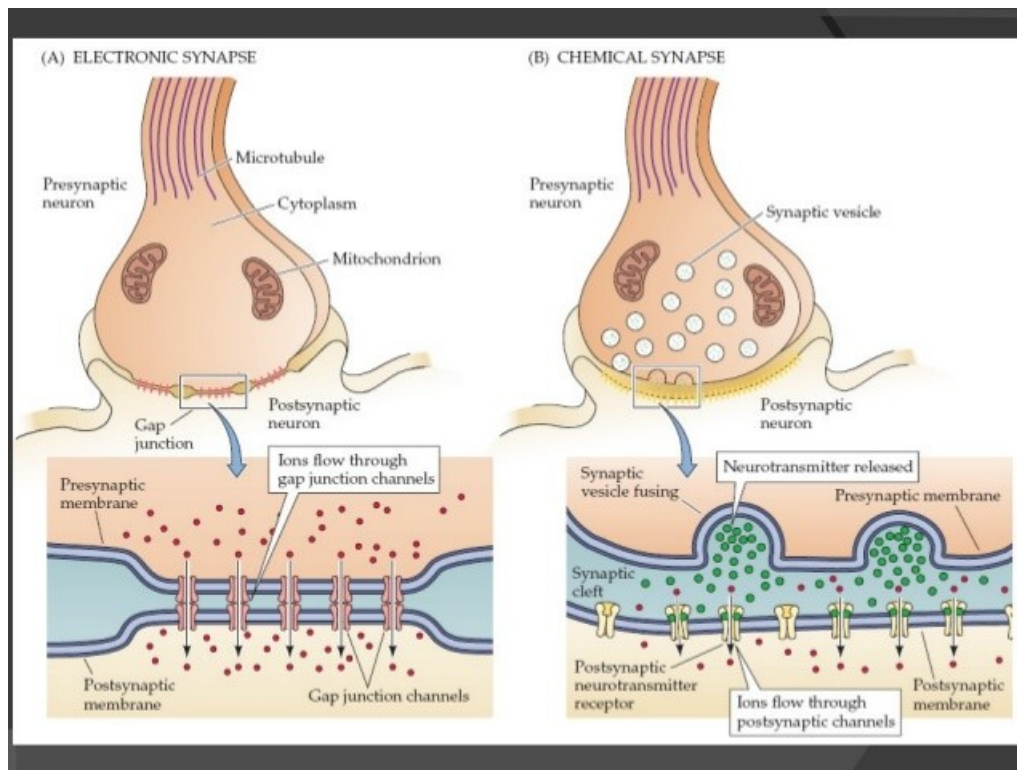


Figura 5: Sinapsis eléctrica y química en las neuronas biológicas. Fuente: www.physicsforums.com

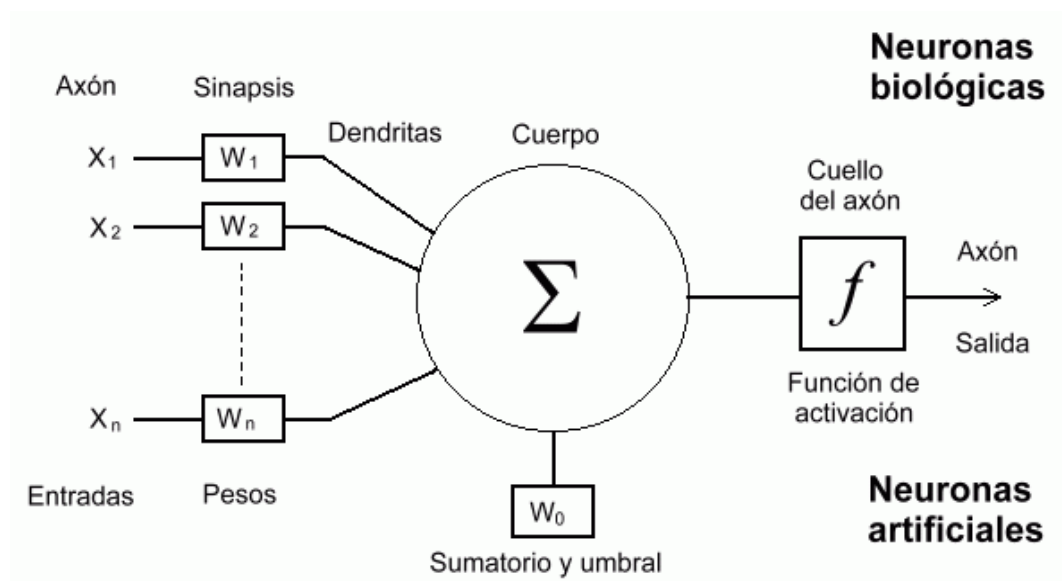


Figura 6: Partes de una neurona artificial y sus correspondientes en la neurona biológica.

Tabla 1: Características de las RNB y de las RNA.

Características	Redes Neuronales Biológicas	Redes Neuronales Artificiales
Conexiones	Excitación e inhibición Eléctricas y químicas (sinapsis; Figura 5) Aleatorias y en gran cantidad (cada neurona a otras $10^3 - 10^4$)	Sólo excitación Solamente eléctricas ²³ Especificadas con precisión (varios órdenes inferior)
Procesamiento	Altamente paralelizado 10^{14} sinapsis	Altamente paralelizado 10^8 transistores
Tamaño de la red	10^{11} neuronas	$10^2 - 10^4$ neuronas
Velocidad de procesamiento	Varios milisegundos + retardos para estabilización química (fase refractaria)	Varios nanosegundos
Almacenamiento de la información	En la sinapsis	En la matriz de pesos
Partes (Figura 6)	Dendritas, cuerpo (soma, pericarion) y axón	Entrada, nodo (unidad, neurona) y salida
Señales	Moduladas en frecuencia (pulsos)	Moduladas en amplitud (numéricas)
Aprendizaje	Basado en hechos pasados, ajustando conexiones sinápticas	Basado en hechos pasados, ajustando pesos
Estilo de computación	Paralelo y distribuido	Paralelo y distribuido

Nótese que, desde hace décadas, la totalidad de las supercomputadoras utilizan algunas características propias de las RNB tales como la computación distribuida y en paralelo, la memoria distribuida e integrada (memorias caché) y la gran cantidad de procesadores (aún así, muchos órdenes de magnitud inferior al número de neuronas biológicas presentes en el cerebro²⁴).

23 Nos referimos a las implementaciones en hardware (e.g. con VLSI, *Dynamic Ring Architecture*, servidores CNAPS de Adaptive Solutions, etc) de las RNA, no simuladas mediante software (McCartor 1991; Cabestany et al. 1996).

24 Según Top500.org, hasta Septiembre 2014 el máximo número de núcleos en un supercomputador en paralelo es de 3.120.000 en el clúster de ordenadores “*MilkyWay-2*” del *National Super Computer Center* en Guangzhou (China). Cuenta con una memoria de 1.375 TiB y requiere una potencia de 17,6 MW; 24 MW contando con la refrigeración.

Tabla 2: Características del computador de Von Neumann original y de las RNB.

Características	Computador de Von Neumann original	Redes Neuronales Biológicas
Procesador	Complejo Alta velocidad Uno o unos pocos	Simple Baja velocidad Gran cantidad (10^{11})
Memoria	Separada del procesador Localizada No direccionable por contenido	Integrada Distribuida Direccionable por contenido
Computación	Centralizada Secuencial Programas almacenados	Distribuida Paralela Autoaprendizaje
Fiabilidad	Muy vulnerable	Robusto
Habilidades destacadas	Manipulaciones numéricas y simbólicas	Problemas perceptivos Basados en comportamiento
Entorno de funcionamiento	Bien definido Bien delimitado	Pobremente definido Sin restricciones

1.4.1 Clasificación de Redes Neuronales Artificiales

(Rumelhart, Hinton y McClelland 1986; Álvarez Romero, Domínguez Brito y Suárez Araujo 2013) definen un marco general (*framework*) con el que caracterizar o modelar una RNA y que se compone de los siguientes aspectos:

- Un conjunto de unidades de proceso y sus respectivas memorias locales. Pueden ser unidades de entrada, salida u ocultas.
- El estado de activación de cada una de las unidades de proceso en un momento dado. Puede ser continuo o discreto y limitados o no a un cierto rango.
- Una función de salida para cada unidad de proceso. Usualmente se trata de una función umbral.
- Un patrón de conectividad entre las unidades de proceso, en el que cada conexión viene acompañada de un peso sináptico que indica la fuerza de la conexión.
- Una regla de propagación o función de red, con la que difundir los patrones de actividades a través de la red de conexiones.
- Una regla o función de activación, para combinar las entradas que llegan a una unidad con el estado actual de dicha unidad para luego producir un nuevo nivel de activación en ésta.
- Una regla de aprendizaje, mediante la cual poder alterar los patrones de conectividad en base a la experiencia. Dicha alteración puede deberse a la creación, modificación (a través de cambios en los pesos sinápticos) o pérdida de conexiones sinápticas.
- Una representación del entorno dentro del cual el sistema debe operar, existiendo un entorno de información local y otro global.

Siguiendo este marco caracterizamos las RNA en los siguientes tres niveles:

- Topología de conexiones: también conocido como el modelo de interconexión entre los diferentes nodos que componen la arquitectura neuronal, también abarcando la estructura neuronal (Miiikkulainen 2010; Álvarez Romero, Domínguez Brito y Suárez Araujo 2013). Ésta se suele describir formalmente a través de un grafo orientado, correspondiéndose sus

nodos con las neuronas o unidades de procesamiento, y sus arcos orientados con los canales de comunicación entre neuronas. Cada arco viene asociado a un valor real que indica el peso de la conexión. El esquema de comunicación entre las distintas neuronas puede ser o bien jerárquico (sólo son posibles conexiones entre niveles o capas²⁵ inmediatamente consecutivas) o bien no jerárquico (las conexiones pueden saltarse niveles intermedios).

- **Neurodinámica:** puede venir expresada mediante funciones tanto continuas en el tiempo como discretas, en cuyo caso hay que indicar cuando se efectúan los cálculos de las salidas de los elementos de proceso y se realizan por tanto los cambios en los valores de salida de éstos. Dichos cambios pueden ser realizados síncrona o asíncronamente, y entre todos o sólo con parte de los elementos de la red. Entre las formas más habituales dentro de los métodos asíncronos nos podemos encontrar con la aleatoria, en la cual se eligen aleatoriamente la unidad de proceso que va a computar su salida. Otra forma muy común es siguiendo el orden topológico impuesto por el conexionado, donde los cómputos se van realizando síncronamente capa a capa, desde las próximas a las entradas hasta la de salida.
- **Aprendizaje:** es la habilidad que posee un sistema para inferir u obtener información del entorno sin requerir que fuera programado externamente. Este término está íntimamente relacionado con el dilema "plasticidad – estabilidad": el sistema debe ser suficientemente adaptativo para aprender nuevo conocimiento (plasticidad) y, al mismo tiempo, no olvidar lo anteriormente aprendido (estabilidad) (Gabrielsson y Gabrielsson 2006). En las RNA el aprendizaje se puede dividir en dos fases o modos de funcionamiento, usualmente bien diferenciadas entre sí en la mayoría de RNA: modo de carga (se realiza el aprendizaje propiamente dicho, es decir, actualizar los pesos de las conexiones en base a la información extraída tras procesar los datos de entrada) y modo de recuperación (a través del conocimiento almacenado y en base a éste, obtener respuestas ante ciertas entradas, presumiblemente, aunque no necesariamente, no conocidas con anterioridad).

Atendiendo a las características anteriores podemos clasificar las RNA de la forma siguiente (Jain, Mao y Mohiuddin 1996):

- **Topología de la red:** dentro de las topologías jerárquicas, se distinguen arquitecturas monocapa (*e.g.*: perceptrón simple) y multicapa, mucho más comunes y versátiles (Gabrielsson y Gabrielsson 2006). Adicionalmente, dentro de las redes jerárquicas multicapa se pueden distinguir subvariantes dependiendo de la manera en la que se interconectan las neuronas de las distintas capas:
 - *Feed-forward*: la información fluye desde la capa de entrada hasta la de salida, no existiendo bucles. Ej.: perceptrón multicapa.
 - *Feedback (recurrent)*: pueden existir bucles. Ej.: red de Hopfield.
- **Mecanismo de aprendizaje:** como ya hemos dicho, en el contexto de las RNA el proceso de aprendizaje consiste en la actualización de los pesos de las conexiones de manera que la red pueda realizar eficientemente una tarea específica. Dependiendo del tipo de guía que reciba la red, podemos distinguir tres grandes grupos:
 - **Supervisado:** utiliza una señal supervisora. Podemos decir que es similar al aprendizaje con un profesor dado que a la red se le proporciona, para cada entrada, la salida correcta o esperada. En base a cada par entrada-salida, la red ajustará su conjunto de pesos basándose en alguno de los algoritmos de aprendizaje disponibles:
 - Por corrección del error: el ajuste de los pesos se basa en reducir el error – local a cada neurona, o global a toda la red – como la diferencia entre la salida deseada y la obtenida por la red. Ejemplos: perceptrón, Bayes, *Support Vector Machine* (SVM), *Learning Vector Quantization* (LVQ), etc.

25 Una capa está conformada por conjuntos de neuronas con funcionamiento, conexionado y tipo de procesamiento similares. Mayoritariamente, se suelen distinguir tres: capa de entrada, capas ocultas, y capa de salida.

- Por refuerzo: se aplica una función de refuerzo – positiva en caso de éxito (recompensa) y negativa en caso contrario (castigo) – en caso de que la salida de la red se ajuste adecuadamente a la deseada.
- Estocástico: a través de cambios aleatorios – ya sea con funciones de transferencia estocásticas o con pesos estocásticos – en los pesos y en la evaluación de su efecto a partir del objetivo deseado y de distribuciones de probabilidad (mediante el uso de funciones de energía como representantes de la estabilidad de la red). Ej.: máquinas de Boltzmann.
- **No supervisado:** categorizan las entradas sin requerir ninguna información sobre la salida deseada, siendo la propia red la que debe encontrar por si misma la regularidad, correlación o patrones subyacentes en los datos de entrada para luego agruparlos, clusterizarlos o categorizarlos en consecuencia. Algunos tipos de algoritmos:
 - Hebbiano: basado en la observación que hizo el psicólogo canadiense Donald O. Hebb durante ciertos experimentos neurobiológicos: *“Supongamos que la persistencia de una actividad repetitiva (o “señal”) tiende a inducir cambios celulares duraderos que promueven su estabilidad. [...] Cuando el axón de una célula A está lo suficientemente cerca como para excitar a una célula B y repetidamente toma parte en la activación, ocurren procesos de crecimiento o cambios metabólicos en una o ambas células de manera que tanto la eficiencia de la célula A, como la capacidad de excitación de la célula B son aumentadas”* (Hebb 1949). Esta observación suele conocerse como “ley o postulado de Hebb” o “teoría del conjunto o asamblea celular”, y resumirse según la frase de la neurobióloga estadounidense Carla J. Shatz: *“las células que se activan juntas, permanecerán conectadas”* (Schatz 1992).
 - Competitivo: las neuronas de la red compiten entre ellas por activarse. Una de las reglas más comunes seguida por las redes competitivas es la denominada como *winner-take-all*: ante un patrón de entrada determinado, solamente una (o un pequeño grupo alrededor de ésta) de ellas, conocida como neurona ganadora, se activa u obtiene el mejor valor. Esta es la regla seguida por los mapas autoorganizados (Haykin 1999, 2009), los cuales usaremos durante este proyecto y que se explicarán profusamente en el apartado 1.4.2. Además de dicha regla, existe la conocida como *winner-take-most* (utilizada por las redes de tipo *neural gas*, más flexibles que los mapas autoorganizados): los pesos sinápticos se adaptan independientemente de la estructura topológica²⁶ de las neuronas dentro de la red, sino según las distancias relativas de las neuronas en el espacio de entrada (Martinetz y Schulten 1991).
- **Semi-supervisado:** parte de los datos de entrada, que cuentan con etiquetas, los aprende de forma supervisada, y la otra parte, carente de etiquetas y típicamente mucho más numerosa, de manera no supervisada (Chapelle, Schölkopf y Zien 2006; Zhu 2008, 2010).

26 En el caso de los mapas autoorganizados, si se desea optimizar tanto la conservación de la topología del mapeado como el uso de todas las neuronas, la topología de la red debe haber sido necesariamente preestablecida. Más aún, ésta debe coincidir con la topología original de los datos, para lo cual se requiere algún tipo de conocimiento previo, algo que no siempre está disponible o puede ser complicado de conseguir con estructuras topológicas originales muy heterogéneas (Martinetz y Schulten 1991).

1.4.2 Mapas autoorganizados de Kohonen

Fueron descritos por primera vez por el investigador y profesor emérito finés Teuvo Kohonen, de ahí que frecuentemente sean denominados redes o mapas de Kohonen (Rojas 1999; Sivanandam y Deepa 2006; Bullinaria 2012a; Leow 2014). Otras posibles denominaciones son las de *SOM* y *SOFM*²⁷, esta última menos común.

Como hemos dicho anteriormente, se trata de una RNA que se entrena mediante aprendizaje no supervisado. Su característica principal es que permite el escalamiento multidimensional, es decir, la transformación de una entrada n -dimensional en una representación discreta (mapa) de entre 1 y 3 dimensiones, siendo la más frecuente la bidimensional. Dicho mapa, como comprobaremos más adelante, cumplirá el principio de formación de mapas topográficos en neurobiología (Haykin 1999; Bullinaria 2012b): “la localización espacial de una neurona de salida en un mapa topográfico se corresponde con un dominio o característica particular extraída del espacio de entrada”, es decir, por un lado, la información entrante estará en todo momento encapsulada en un contexto o vecindad apropiado y, por otro lado, las neuronas que manejen información similar estarán próximas entre sí (con el fin de que puedan interactuar vía conexiones sinápticas cortas). Basándose en la observación anterior se ha podido declarar que ciertas áreas (conjuntos de neuronas cercanas) del cerebro humano están especializadas en funciones concretas (Figura 7).

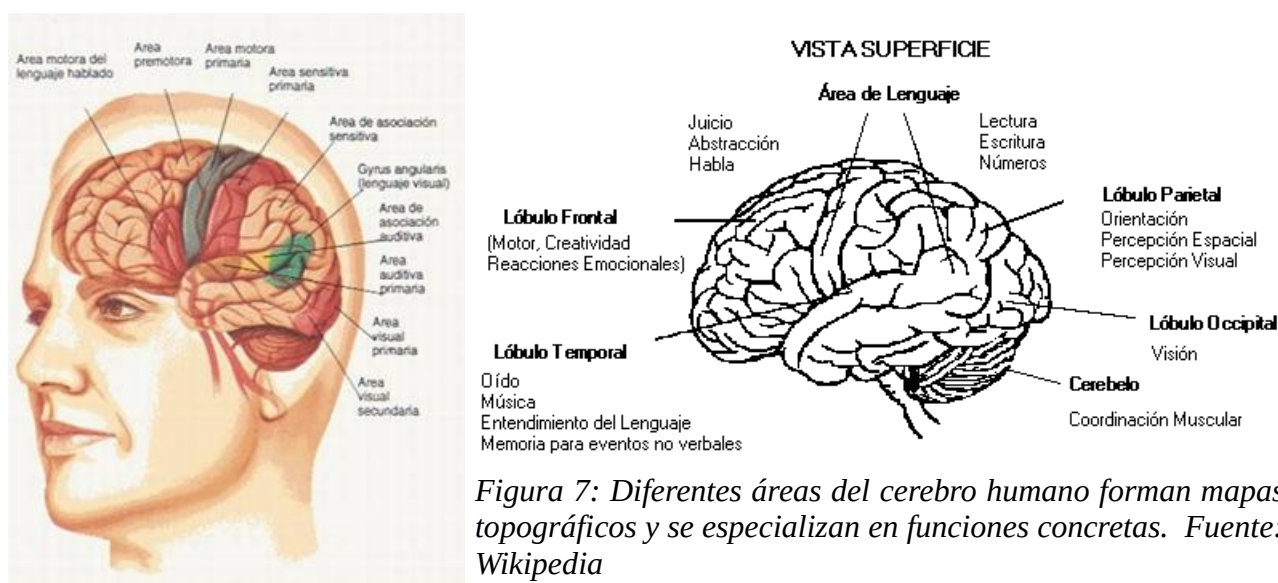


Figura 7: Diferentes áreas del cerebro humano forman mapas topográficos y se especializan en funciones concretas. Fuente: Wikipedia

La estructura de un *SOM*, de apariencia similar a una red *feedforward* aunque bastante distinto en su funcionamiento, está conformada por dos capas de nodos o neuronas (Leow 2014):

- Entrada: cada neurona de esta capa, que representa a un atributo o característica concreta del vector de datos de entrada, está conectada con todas las neuronas de la capa siguiente.
- Competitiva o de Kohonen: los nodos dentro de esta capa, los cuales no están interconectados, tienen asociados tanto un vector de pesos – de igual tamaño que el número de neuronas en la capa de entrada – como una posición dentro del mapa, dentro del cual las neuronas están dispuestas regularmente en una rejilla con forma rectangular o hexagonal.

Esta RNA, como ya se ha dicho que ocurre en todas las RNA, opera o bien en modo

²⁷ Siglas de las palabras en inglés *Self-Organizing Maps* y *Self-Organizing Feature Maps*, respectivamente.

entrenamiento (aprendizaje) o bien en modo carga (o mapeo en este caso). Durante el proceso de aprendizaje se llevará a cabo la construcción del mapa topográfico mediante un proceso competitivo sobre el conjunto de entrenamiento (Hill 1999). En el proceso de mapeo se realizará la asignación automática de nuevos vectores de entrada en el mapa generado. Centrándonos en el modo aprendizaje del SOM, podemos describir las distintas etapas que conforman el mismo y por ende el algoritmo de aprendizaje seguido (Haykin 1999; Sayad 2010; Bullinaria 2012b; Wikipedia 2013g; Leow 2014):

1. Inicialización: cada vector de pesos $\mathbf{w}_j$ de cada uno de los nodos j de la capa de competición son inicializados con valores aleatorios²⁸ reducidos, comprendidos en el intervalo (0, 1).
2. Competición: para cada patrón o vector de entrada $\mathbf{X}$, las neuronas compiten entre sí, computando el valor de una función distancia (discriminante) dada, entre dicho vector y el vector de pesos de la neurona en cuestión. La distancia más utilizada suele ser la distancia euclídea o euclidiana, Eq. (1). La neurona ganadora, o BMU ²⁹, $\mathbf{c}(\mathbf{X})$ es la que consigue un valor menor *ergo* su vector de pesos $\mathbf{w}_{c(X)}$ es el más similar o cercano a ese vector de entrada $\mathbf{X}$ de entre todos los de las demás neuronas de la capa de Kohonen (Eq. (2)).

$$\|\mathbf{W}_j - \mathbf{X}\| = \sqrt{\sum (w_{ji} - x_i)^2} \quad (1)$$

$$c(X) = \arg \min_j \|\mathbf{W}_j - \mathbf{X}\| \quad (2)$$

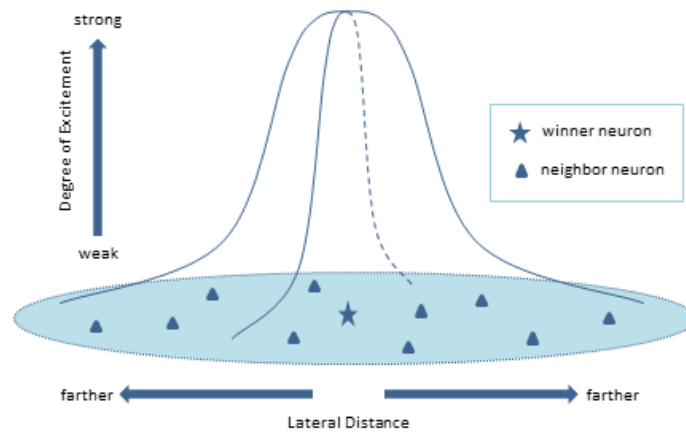


Figura 8: Vecindad topológica con forma de función gaussiana, indicando el grado de excitación de las neuronas según la distancia lateral con la ganadora. Fuente: (Leow 2014)

3. Cooperación: la BMU define la ubicación espacial de una vecindad topológica (Figura 8), formada por la propia neurona ganadora y otras adyacentes también excitadas (debido a un proceso que en neurobiología se llama interacción lateral). Esta vecindad decae (Figura 9) tanto con la distancia (normalmente con forma de función gaussiana: valor máximo en la BMU y va decreciendo simétrica y gradualmente hasta 0 con reducciones en la distancia) como con el tiempo (usualmente exponencial negativa; al principio la interacción entre nodos es casi global y, con el paso del tiempo, será solamente entre neuronas a nivel local).

28 Suele ocurrir que, con el fin de acelerar el aprendizaje del SOM, se decida inicializar éste seleccionando aleatoriamente ejemplares dentro del conjunto de entrenamiento.

29 Siglas de la palabra en inglés *Best Matching Unit*, sinónimo de *Best Matching Neuron* (BMN). Se trata del vector prototipo que posee mayor similitud (es decir, menor distancia) con el dato.

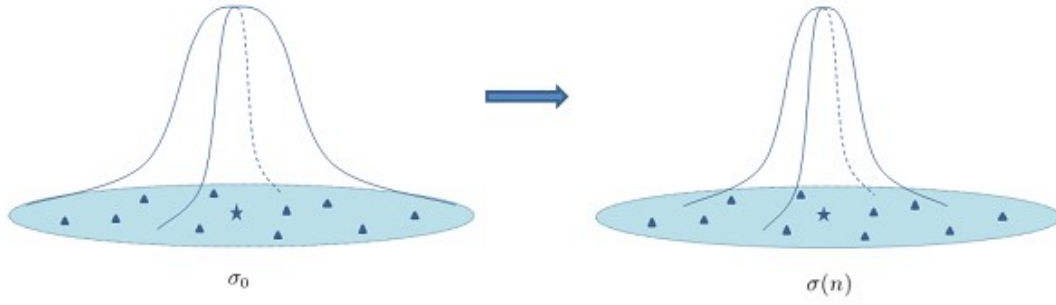


Figura 9: Reducción de la vecindad topológica con el paso del tiempo. Fuente: (Leow 2014)

Sea $h_{j,c(x)}$ la función de vecindad (una medida del grado de excitación de la neurona ganadora $c(\mathbf{X})$ en la neurona $\mathbf{j}$; Eq. 3), $d_{j,c(x)}$ la distancia euclidiana entre dichas neuronas, y $\sigma(n) = \sigma_0 \exp(-\frac{n}{\tau_1})$, $n = 0, 1, 2, \dots$, $\tau_1 = \frac{N}{\log \sigma_0}$ el ancho de la rejilla, con σ_0 como el ancho inicial, τ_1 la constante de tiempo y N el número total de iteraciones en la primera fase del proceso de adaptación,

$$h_{j,c(x)}(n) = \exp\left(\frac{-d_{j,c(x)}^2}{2\sigma(n)^2}\right) \quad (3)$$

Por otra parte, existen diversos tipos de vecindad, distinguiéndose tanto según la forma de la rejilla como según la disposición de los nodos que se consideran vecinos por lo que, en base a ambas características, el número de nodos que conforma una vecindad es diferente. Suponiendo que la forma del mapa sea una hoja en 2-D (muy frecuente) y si sólo tenemos en cuenta a los nodos inmediatamente más cercanos, podemos diferenciar los siguientes tipos, entre otros posibles (Figura 10 y Figura 11) (Westerlund 2005):

- Rejilla hexagonal: 6 neuronas vecinas. Este tipo no favorece tanto las direcciones horizontal y vertical como ocurre con las rejillas rectangulares (Kohonen 2001; Mikkelsen 2006).
- Rejilla rectangular:
 - Vecindad de Von Neumann: 4, correspondientes a los nodos ortogonales.
 - Vecindad de Moore: 8, incluyendo tanto ortogonales como los que no lo son.

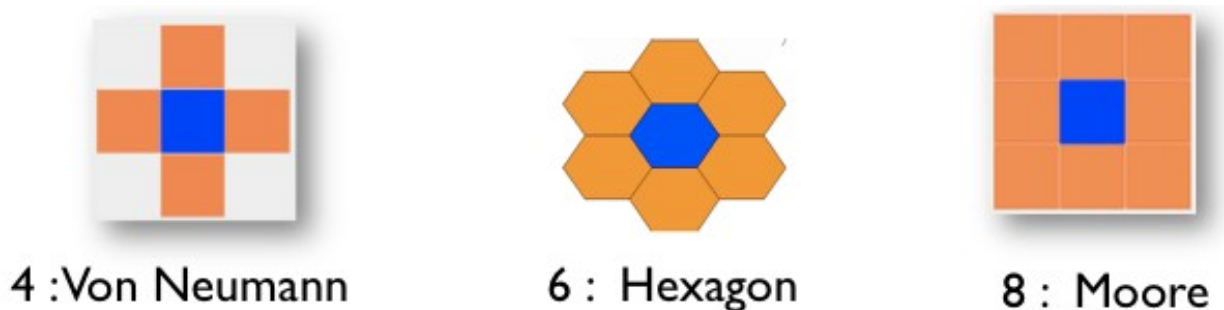


Figura 10: Diferentes tipos de vecindad: Von Neumann (4 vecinos), hexagonal (6) y Moore (8). Fuente: Gama Platform Grid Species (code.google.com)

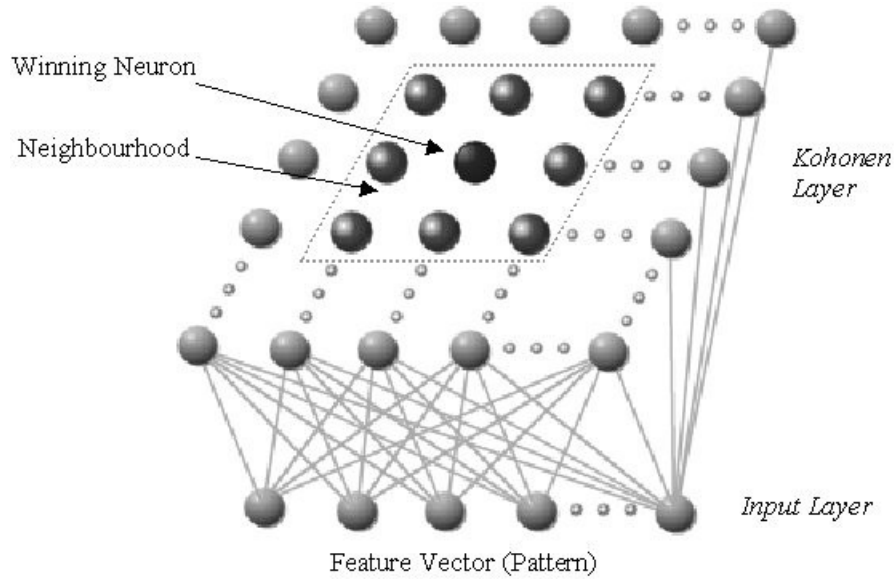


Figura 11: Los conceptos de *BMU* y *vecindad (de Moore)* sobre la estructura de un *SOM (2-D)*. Fuente: Richard M. Clark (NTL World).

4. **Adaptación:** las neuronas excitadas, *BMU* $\mathbf{c}(\mathbf{X})$ y sus vecinas, son las que adaptan sus pesos. Ellas decrementan sus valores individuales de la función distancia usada, en relación al patrón de entrada $\mathbf{X}$, a través de ajustes apropiados en sus respectivos vectores de pesos $\mathbf{w}_j$ de manera que dicha *BMU* y adyacentes respondan más adecuadamente ante futuros patrones de entrada iguales o similares. Estos ajustes en los vectores de pesos, como hemos visto en la etapa precedente, no son iguales para cada una de la neuronas excitadas en tanto en cuanto la ganadora se actualizará en mayor medida que sus vecinas. Con ello se consigue acercar el vector de pesos de la *BMU* al vector de entrada lo que, tras sucesivas repeticiones en la presentación de los datos de entrada, permitirá conseguir una ordenación topológica, el mapa topológico. Para asegurar la convergencia del *SOM*, el ratio de aprendizaje debe decrecer (exponencial negativa suele ser) con el tiempo, así como la vecindad. Los nuevos pesos se calculan de la forma indicada en la Eq. 4.

$$\mathbf{w}_j(n+1) = \mathbf{w}_j(n) + \eta(n) \cdot h_{j,c(x)}(n) \cdot (\mathbf{x} - \mathbf{w}_j(n)) \quad (4)$$

Siendo $h_{j,c(x)}(n)$ la función de vecindad entre la neurona $\mathbf{j}$ y la ganadora $\mathbf{c}(\mathbf{X})$ en la iteración $\mathbf{n}$ (Eq. 5) y $\eta(n)$ el ratio de aprendizaje, variable en el tiempo, también en la iteración $\mathbf{n}$, con η_0 como el ratio de aprendizaje inicial, y τ_2 una constante de tiempo cuyo valor suele ser el número total de iteraciones en la primera fase del proceso de adaptación N

$$\eta(n) = \eta_0 \cdot \exp\left(-\frac{n}{\tau_2}\right) , \quad n = 0, 1, 2, \dots, N \quad (5)$$

5. **Iteración:** se repite desde el paso 2 de este algoritmo de aprendizaje el suficiente número de iteraciones con el fin de conseguir la convergencia, que puede que no se llegue finalmente a conseguir, especialmente con redes de Kohonen con dimensión mayor a la unidad (Erwin, Obermayer y Schulten 1991, 1992). Ello suele ocurrir al "estancarse" el *SOM* en un estado metaestable³⁰ (Figura 12), donde la convergencia es pobre. Para evitarlo, hay que tener

³⁰ Dícese del estado que, aún siendo estable, lo es débilmente. En nuestro caso, indica que el mapa sigue desordenado

especial precaución con la elección del número de iteraciones y de la función de vecindad (Rojas 1999; Bullinaria 2012b), entre otros.

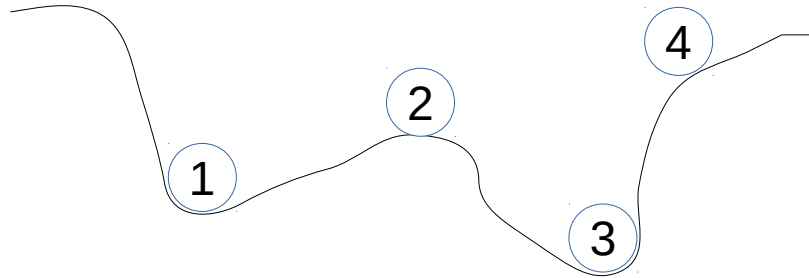


Figura 12: Estabilidad de los estados: metaestable (1), estable o de mínima energía (3) e inestable (2 y 4).

Es relevante mencionar que el algoritmo anteriormente explicado es el conocido por los nombres de "secuencial", "incremental", "patrón a patrón" u "*online*", entre otros muchos (Fort, Letremy y Cottrell 2002; Wilson y Martinez 2003; Gabrielsson y Gabrielsson 2006). Es el mayormente utilizado a nivel académico por su simplicidad. La principal diferencia con la otra variante que expondremos a continuación es que, como hemos visto, los pesos sinápticos se actualizan tras la presentación de cada vector de entrada, o dicho de otra forma, tras cada iteración, paso o instancia.

La otra versión de algoritmo de aprendizaje es la denominada "*batch*", en la cual los pesos se actualizan tras la presentación de todos los vectores de entrada, es decir, tras cada ciclo o *epoch*³¹, de ahí que dicho algoritmo sea también conocido como entrenamiento "basado en *epochs*" (Gabrielsson y Gabrielsson 2006). Para ello, se acumulan, en una lista para cada neurona del SOM, los vectores de entrada cuya *BMU* fuera dicha neurona. Dicha acumulación es temporal hasta que sean finalmente utilizadas para actualizar los pesos al final de cada *epoch*. Sobre cada lista se aplica la media, o genéricamente la mediana generalizada (Kohonen 2001, 2013), teniendo en cuenta la vecindad tratada en esa *epoch*. Los nuevos pesos se hallan de la manera señalada en la Eq. 6 (Mikkelsen 2006). Es bastante común el uso del algoritmo *batch* en casi todos los ámbitos debido a su rapidez de convergencia, buen hacer y capacidad de utilizar datos no vectoriales, a costa de un alto consumo de memoria (Kohonen 2001, 2013).

$$w_j(n+1) = \frac{\sum_i n_i \cdot h_{ji} \cdot \bar{x}_j}{\sum_i n_i \cdot h_{ji}} \quad (6)$$

siendo $\bar{x}_j$ la media de los vectores de entrada en las listas recogidas durante esa *epoch*; n_i el número de elementos en la lista, y ambos sumatorios se aplican sobre la totalidad de las neuronas de la red.

y que se podría mejorar y conseguir mayor estabilidad (en muchos campos, el estado más estable se denomina "estado de mínima energía"; ver Figura 12).

31 Conjunto finito de entradas-con-significado presentadas secuencialmente a la red. En concreto, relativo al algoritmo en su versión *batch*, suele considerarse como sinónimo de "ciclo", es decir, presentación de la totalidad del conjunto de entrenamiento a la RNA para su aprendizaje, tras la cual se aplican las actualizaciones en los pesos acumulados (Wilson y Martinez 2003). No confundir con "instancia" o presentación de un único patrón de entrada.

1.4.3 Medición de la calidad

Hemos implementado dos maneras de concebir la medición de la calidad de los clasificadores, de las muchas posibles (Villmann, Der y Martinetz 1994a, 1994b ; Villmann et al. 1994; Kaski y Lagus 1996; Kiviluoto 1996; Pözlzbauer 2004; Sokolova, Japkowicz y Szpakowicz 2006; Álvarez Romero, Domínguez Brito y Suárez Araujo 2013), las cuales se diferencian, principalmente, en las características que miden:

- Calidad del mapa: son independientes de que originalmente existan etiquetas en los datos utilizados.
- Rendimiento del clasificador: no serían aplicables si no poseyéramos datos etiquetados con los valores esperables y correctos que los SOM deberían retornar.

1.4.3.1 Calidad del mapa

Por un lado, para obtener los resultados expuestos a lo largo del Capítulo 3, tanto sobre el conjunto de entrenamiento-ajuste como sobre el de testeo, se han utilizado las siguientes dos formas de medición de la calidad (Westerlund 2005; Uriarte y Martín 2005; Sassi 2011) que ya vienen implementadas en la *SOM Toolbox* mencionada:

- Error de cuantización medio (*MQE*)³²: señala la distancia media entre cada vector de datos y su *BMU*. Mide la resolución del mapa. Una configuración de *SOM* con menor *MQE* puede ser considerada mejor que otra con un valor mayor.

$$QE = \frac{1}{N} \sum_{i=1}^N d(x_i, w_{x_i}) \text{ y luego calculando su media } MQE = \frac{1}{c} \sum_{j=1}^c QE_j$$

siendo $d(x_i, w_{x_i})$ la distancia entre el vector de datos x_i y su *BMU* w_{x_i} , N el número de vectores de entrada, y c la cantidad de neuronas ganadoras.

- Error topográfico (*TE*)³³: indica la proporción de todos los vectores de datos para los cuales la primera y segunda *BMU* no son unidades adyacentes. Mide la preservación de la topología. Los valores devueltos están en el rango $[0, 1]$, considerándose el valor de 0 como la preservación topológica perfecta, es decir, la mejor posible.

$$TE = \frac{1}{N} \sum_{k=1}^N u(x_k) \text{ donde } u(x_k) = \begin{cases} 1, & \text{si la 1ª y la 2ª BMU no son adyacentes} \\ 0, & \text{en caso contrario} \end{cases}$$

siendo N el número de vectores de entrada.

Según la documentación de la *SOM Toolbox*, ambas medidas dan los mejores resultados cuando el mapa ha sobreajustado los datos³⁴, lo que suele pasar cuando el número de neuronas es igual o mayor que el número de ejemplos de entrenamiento, cuando también la resolución y la preservación de la topología se convierten en objetivos contradictorios.

Si nos atenemos a (Pözlzbauer 2004), *MQE* puede fácilmente reducirse simplemente aumentando el número de nodos de la red ya que los datos se distribuyen más espaciados por el mapa, pero ello, normalmente, viene acompañado del efecto secundario indeseado de distorsionar – *ergo*, empeorar – la topología del mapa, o sea, aumentar el *TE*. Sobre esta última medida aduce que no es aconsejable para mapas pequeños y, por otra parte, que se ve penalizada cuando el mapa tiene unas dimensiones demasiado e innecesariamente grandes para el conjunto de datos. De este autor

³² Del inglés *Mean Quantization Error*.

³³ Del inglés *Topographic Error*.

³⁴ También es frecuente emplear su versión inglesa: *data overfitting*.

inferimos que, en conjunto, los mejores resultados se obtendrán cuando la *TE* obtiene un mínimo (gráficamente sería posiblemente en una función cóncava) acompañado por un bajo *MQE*.

1.4.3.2 Rendimiento del clasificador

Por otro lado, hemos de mencionar el otro conjunto de métodos de evaluación (Bradley 1997) de, usualmente, clasificadores binarios³⁵, más exhaustivo y realista que los anteriores *MQE* y *TE* al tener en cuenta las etiquetas asociadas con los datos, como ya habíamos mencionado:

- Falsos Positivos (*False Positive*, *FP*, falsa alarma o error tipo I): cuando correos legítimos fueron identificados incorrectamente como no deseados. Este tipo de errores es el que se debe evitar a toda costa ya que la mayoría de usuarios no revisan la carpeta de correo no deseado en busca de posibles errores de clasificación con correos electrónicos que sí desearían recibir (Wiehes 2005). No confundir con los Verdaderos Positivos (*True Positive*, *TP*): spam correctamente categorizado como tal.
- Falsos Negativos (*False Negative*, *FN*, o error tipo II): cuando correos no deseados fueron clasificados incorrectamente como legítimos. En la mayoría de los escenarios, se puede ser algo más flexible con esta clase de errores que con los *FP*, siempre y cuando se mantengan en bajo número (Wiehes 2005). No confundir con los Verdaderos Negativos (*True Negative*, *TN*): ham clasificado como tal.
- Precisión (Wikipedia 2013e, 2014a): o *Positive Predictive Value (PPV)*. Es una medida de la calidad o la proporción de spam correctamente identificado sobre todos los correos electrónicos que el filtro ha señalado como no deseados.

$$\text{Precisión} = \frac{TP}{TP + FP}$$

- Sensibilidad (Wikipedia 2013h, 2013e): también *recall* o *True Positive Rate (TPR)*. Mide la completitud, la cantidad o la habilidad de identificar spam del filtro, es decir, la proporción de spam correctamente clasificados como tal sobre todos los correos electrónicos que sí son no deseados. Alta sensibilidad lleva aparejada una ratio baja de Falsos Negativos.

$$\text{Sensibilidad} = \frac{TP}{TP + FN}$$

- Especificidad (Wikipedia 2013e, 2013c): o *True Negative Rate (TNR)*. Indica cuán hábil es el filtro en la clasificación de correos legítimos. Alta especificidad va acompañada de una baja ratio de Falsos Positivos.

$$\text{Especificidad} = \frac{TN}{TN + FP}$$

- Exactitud (Bradley 1997; Wikipedia 2013e, 2013c): o *accuracy*. Indica lo cerca que el valor medido se encuentra del valor real, es decir, la fracción de correos electrónicos correctamente categorizados. (Metz 1978; Provost, Fawcett y Kohavi 1997) no la recomiendan, especialmente si se desconocen la distribución de las clases y los costes de los errores, prefiriéndose curvas *ROC* en tal caso.

$$\text{Exactitud} = \frac{TP + TN}{TP + FP + TN + FN}$$

- Matriz de confusión (Swets 1988; Powers 2011): o tabla de contingencia (Tabla 3).

35 Sistemas que realizan una clasificación binaria o binomial, es decir, separan unos elementos de otros en dos grupos según cumplan o no cierta propiedad (Wikipedia 2013c).

Representa visualmente el rendimiento de un clasificador, siendo las filas las clases predichas o diagnosticadas, y las columnas las clases verdaderas o actuales³⁶. Podemos afirmar que la mejor clasificación viene indicada por el cumplimiento simultáneo de las dos siguientes propiedades: diagonal principal máxima y diagonal secundaria mínima, es decir, la totalidad de las predicciones fueron correctas, y nulas las incorrectas. En forma de ecuaciones, en el clasificador perfecto se cumpliría, siendo N el número de patrones o vectores de entrada:

$$TP+TN = N$$

$$FP+FN = 0$$

Tabla 3: Matriz de confusión o tabla de contingencia: TP , FP , TN y FN .

Matriz de Confusión	CONDICIÓN		
		spam	ham
	PREDICCIÓN		
	spam	TP	FP
	ham	FN	TN

- F -score (Wikipedia 2013e): F_1 score o F -measure. Indica la exactitud de las clasificaciones realizadas, siendo mejor cuanto más cerca esté esta puntuación de la unidad. (Powers 2011) recomienda utilizarla conjuntamente con otras métricas ya que ésta ignora los TN .

$$F\text{-score} = 2 \cdot \frac{\text{Precisión} \cdot \text{Sensibilidad}}{\text{Precisión} + \text{Sensibilidad}} = 2 \cdot \frac{TP}{2 \cdot TP + FP + FN}$$

- Curva ROC ³⁷ (Metz 1978; Bradley 1997; Duda, Hart y Stork 2001; Forcada 2003; Worster, Fan y Upadhye 2006; Brown y Davis 2006; Slaby 2007a, 2007b ; Lobo, Jiménez-Valverde y Real 2008; Airola et al. 2009; López et al. 2012; Indrayan 2012; García Abad 2012; Álvarez Romero, Domínguez Brito y Suárez Araujo 2013; Navan 2014; Tape, University of Nebraska Medical Center 2014; Systat Software 2015): indica gráficamente el rendimiento de un clasificador, siendo el eje Y la sensibilidad y el eje X el FPR ³⁸. Es por ello que es capaz de encapsular toda la información contenida en la matriz de confusión ya que FN es el complemento de TP , y TN es el complemento de FP (Swets 1988). El clasificador será mejor³⁹ cuanto más cerca pase su curva ROC al punto de clasificación perfecta en (0, 1). Dicho punto – donde su AUC ⁴⁰ es máxima, es decir, vale la unidad – se corresponde con el que obtendría el test *gold standard*, caracterizado por carecer tanto de FN (100% sensibilidad) como de FP (100% especificidad) (Zweig y Campbell 1993). La curva ROC se caracteriza por ser independiente tanto de la distribución de las clases como de los costes de los errores (Provost, Fawcett y Kohavi 1997). En la Figura 13 podemos observar que el clasificador discreto C' es el mejor de los cuatro presentes y fue construido simplemente

36 Algunos autores utilizan la versión traspuesta de dicha matriz. Ambas se pueden usar indistintamente por ser igual de válidas.

37 En el campo de la detección de señales es el acrónimo de la palabra inglesa *Receiver Operating Characteristic* o Característica Operativa del Receptor, mientras que en aplicaciones generalizadas lo es de *Relative Operating Characteristic* o Característica Operativa Relativa (Swets 1988).

38 De *False Positive Rate*, Ratio de Falsos Positivos o ratio de falsa alarma.

$$FPR = \frac{FP}{FP+TN} = 1 - \text{Especificidad}$$

39 Esta relación también es conocida como dominancia i.e. "el clasificador dominará a otros [...]" (Slaby 2007a).

40 Siglas de la voz inglesa *Area Under the ROC Curve*, Área Bajo la Curva ROC o, simplemente, Área Bajo la Curva. Está íntimamente relacionada con el concepto matemático de integral, siendo la forma más sencilla de calcularla a través de integración trapezoidal (Bradley 1997). Puede tener un valor comprendido en el rango $0 \leq AUC \leq 1$

invirtiendo las salidas devueltas por el clasificador, consistentemente pobre, C (Fawcett 2004, 2006 ; Wikipedia 2013f). El B se sitúa sobre la línea de decisión aleatoria o no discriminante.

Cabe mencionar que, adicionalmente, se ha implementado un método propio, sustitutivo del *AUC*, que está fundamentado en la menor distancia euclidiana al punto de clasificación perfecta en (0, 1) aunque se ha preferido y dado más importancia a los resultados provenientes del *AUC* por ser ésta una métrica muy común en la bibliografía consultada.

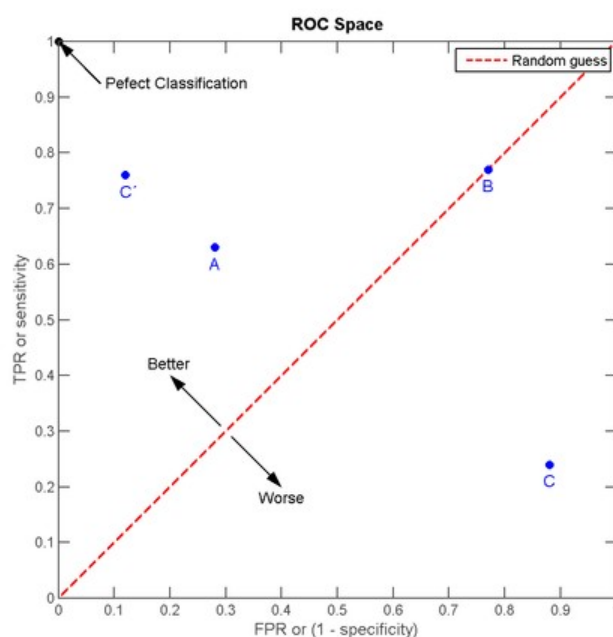


Figura 13: Espacio ROC, línea de decisión aleatoria y 4 clasificadores discretos: A, B, C y C'.
Fuente: Wikipedia.

1.5 Estado de la cuestión

El filtrado antispam se puede realizar en cualquiera de los distintos niveles de la jerarquía implicada en la transmisión de correos electrónicos (Wiehes 2005; Wikipedia 2012a):

- Usuarios finales: tanto emisores como receptores de los mensajes.
- Administradores de sistemas de correos electrónicos.
- Remitentes de correos electrónicos (*email senders*): grandes empresas, normalmente, que se encargan de mandar los mensajes de los usuarios y clientes dentro de su dominio de correo.
- Investigadores policiales y agentes del orden público.

Este análisis del estado de la cuestión, apartado usualmente llamado “estado del arte”⁴¹, está orientado a comprender los diferentes filtros antispam que se han creado, las diferentes técnicas aplicadas, la información sobre la que realizan dicho filtrado, y el grado de necesidad de

⁴¹ Tal y como indica (RAE 2005), se trata de un calco censurable del inglés y deben utilizarse otros términos como "estado/situación actual", "últimos avances" o "estado de la cuestión", según los casos.

supervisión humana antes y durante el entrenamiento del sistema. Nos centraremos en métodos que pueden ser útiles para los dos primeros grupos ya que los dos restantes hacen un mejor uso de otras técnicas (Cabrera y Acosta 2011; Wikipedia 2012a) tales como:

- Para los remitentes de correos electrónicos: el bloqueo del puerto 25 del protocolo *SMTP*, la superación de umbrales de envío de mensajes, condiciones de uso y licencias bastante restrictivas y poco permisivas con el spam, y el uso del *CAPTCHA*⁴² para limitar o ralentizar la creación de nuevas cuentas de correo.
- Para los investigadores policiales y agentes del orden público: la legislación, las multas y la investigación de las webs publicitadas en los spam.

Como veremos a continuación, los sistemas de filtrado antispam que pueden ser utilizados directamente por los usuarios finales y por los administradores de sistemas de correos electrónicos se pueden diseñar y desarrollar atendiendo a diversos aspectos relacionados bien con la fuente de información (de la que obtienen las características necesarias para discernir entre un spam y un ham), bien con el método de diseño (si existe o no algún método de aprendizaje automático). Además de las anteriores técnicas antispam para sendos grupos, existen otras muchas (Wikipedia 2012a) pero de las que difícilmente se pueden construir filtros antispam, de ahí que no se expliquen en profundidad, debido a que se basan en:

- Recomendaciones en las acciones de los usuarios: *e.g.* no hacer públicas direcciones de correo, reportar spam, utilizar cuentas de correo temporales, etc.
- Requerir el uso de hardware adicional como servidores trampa o extremadamente lentos.
- Comprobación de sumas de validación.
- Chequeo de cierre correcto de la conexión *SMTP* mediante el comando *QUIT* (Klensin 2008).

En resumidas cuentas, sólo expondremos los dos siguientes conjuntos distintos, no disjuntos, de formas en las que los sistemas de filtrado antispam se han desarrollado o implementado: según la fuente de información, y según el método de diseño.

1.5.1 Clasificación según la fuente de información

Como pudimos ver en el apartado 1.3.1, todo mensaje de correo electrónico se compone de tres partes – sobre, cabecera y cuerpo – las cuales podemos utilizar como fuente de información para la detección de spam. Este grupo podemos dividirlo en tres: los que se basan en el contenido del correo electrónico (comúnmente hacen uso tanto de la cabecera como del cuerpo), la retroalimentación del usuario (principalmente sobre el cuerpo aunque también partes de la cabecera como la línea del asunto), y la información relativa al sistema (primordialmente, hacen uso de los campos de la cabecera aunque también chequean el sobre, y el cuerpo del mensaje para detectar si está vacío o si contiene archivos adjuntos).

1.5.1.1 Contenido del correo electrónico

Es el más utilizado y podemos encontrarnos con filtros que hacen uso de todo el contenido del mensaje (Cormack y Mojdeh 2009) o, una opción mucho más popular, los que eligen solamente algunas características, en cuyo caso realizan un proceso de selección de éstas:

42 *Completely Automated Public Turing test to tell Computers and Humans Apart* o prueba de Turing Completamente Automática y Pública para diferenciar Ordenadores de Humanos; es una prueba desafío-respuesta utilizada para determinar si el usuario es o no humano, usualmente mediante la utilización de una imagen distorsionada, conformada por un conjunto de caracteres de longitud variable, mostrada por pantalla.

- Basados en firmas de los mensajes y funciones de *hash* robustas (Kolcz, Chowdhury y Alspector 2004).
- Creación de plantillas:
 - Existencia de partes “ancladas”/invariables (Pitsillidis et al. 2010).
 - Existencia de campañas o conjuntos de spam con similares características y, normalmente, mandados en períodos de tiempo cercanos (Qian et al. 2010).
- Basados en reglas o tests (Malathi 2011).
- Utilización de métodos probabilísticos: esencialmente usando clasificadores Bayesianos (Sahami et al. 1998; SpamBayes 2013).
- Uso de información no textual tales como imágenes (Fumera, Pillai y Roli 2006; Gao, Yan y Choudhary 2009; Wikipedia 2012b).
- Combinaciones de varios métodos (Khorsi 2007; The Apache SpamAssassin Project 2013). Especial mención al filtro CRM114 por tener, en alguna de sus variantes, un 99,9% de exactitud con un ratio de falsos positivos inferior al 6% (Yerazunis et al. 2010).

1.5.1.2 *Feedback* o retroalimentación del usuario

En este subgrupo son los propios usuarios del filtro quienes realizan la clasificación manualmente, no habiendo aprendizaje ni autónomo ni automático alguno (Graham-Cumming 2006).

Suele ser considerada la técnica más fiable y robusta (fundamentalmente ante ocultaciones y errores premeditados dentro del propio mensaje) pero algunos estudios, como el anterior, demuestran que presenta errores de clasificación cercanos al 3%. Ello es debido a la similitud, en ciertos casos, entre ham y spam (los denominados *hard* ham (Anderson 2006)) o a la existencia de “casos grises” (James Bruce 2012), donde la categorización tiene una gran componente subjetiva.

1.5.1.3 Información relativa al sistema

Estos clasificadores se aprovechan, por un lado, de las mayores dificultades que tienen los *spammers* ante la modificación de las cabeceras de los correos electrónicos, las cuales son poco propensas a cambios válidos o difícilmente detectables en comparación con el contenido de los cuerpos (Ramachandran y Feamster 2006). Y, por otro lado, de la detección de ficheros adjuntos, especialmente de tipo imagen o ejecutable, estén o no acompañados de cuerpos de mensaje vacíos.

Esta opción es bastante recomendable para los filtros a nivel de remitentes de correos electrónicos aunque, y por eso lo hemos incluido aquí, también puede serlo para los administradores de sistemas de correo electrónico. La razón de esto es que es una metodología con gran capacidad y eficacia, aún trabajando con grandes volúmenes de correos electrónicos, especialmente para las dos siguientes tareas:

- Detección de *botnets*⁴³, siendo un indicativo bastante fiable de su presencia el aumento repentino e inexplicable del envío de correos electrónicos, siendo también detectado por el propio usuario de la máquina con la ralentización de la misma.
- Filtrado por países o continentes porque, si no se espera recibir mensajes desde ciertas regiones, sería interesante y posible bloquear cualquier mensaje entrante proveniente de

43 Redes de ordenadores zombis, usualmente de ámbito internacional, las cuales son explotadas no sólo para mandar spam sino también para el lanzamiento de ataques coordinados contra otras máquinas o sistemas, y para la distribución de software nocivo o dudoso. Suelen estar controladas de manera centralizada por un usuario malintencionado y con grandes conocimientos técnicos.

éstas. Lamentablemente, estos bloqueos, muy utilizados por los primeros filtros antispam (Lutus 2006), ya han sido “driblados” por los *spammers* mediante el uso de las ya mencionadas *botnets*. Desde hace años, EEUU y Asia, respectivamente, son los mayores generadores de correos no deseados aunque se detecta un repunte acusado en algunos países emergentes como la India, Vietnam, Argentina y Brasil (Trustwave 2013a).

Frecuentemente, en este grupo se utiliza la siguiente información para la detección de spam:

- Campos de los propios paquetes donde se mandan los mensajes de correo electrónico (Ramachandran y Feamster 2006), campos provenientes de los protocolos TCP/IP, UDP, SMTP, entre otros. Usualmente se chequean los que incluyen el puerto y la dirección IP, ambos del emisor, puesto que eso ayuda a tomar medidas, sobre todo legales, contra el *spammer*.
- Horas de envío porque es muy poco frecuente enviar mensajes a altas horas de la noche, máxime si son en grandes cantidades. Como ya mencionamos, esto es considerado como una prueba de la pertenencia a una *botnet*.
- Incluir ficheros adjuntos, especialmente si son ejecutables, estén o no comprimidos. La razón es que estos suelen ser el vector de propagación de otras amenazas tales como virus y troyanos, los cuales también podrían ser utilizados para crear una *botnet* y mandar spam.
- Cuerpos carentes de texto y formados solamente por imágenes, es decir, el conocido como spam basado en imágenes (Wikipedia 2012b). Este método fue introducido por los *spammers* para sortear los filtros, mucho más comunes, basados en contenido textual. Suelen hacer uso de algún sistema OCR⁴⁴ lo que ha obligado que los *spammers* hayan decidido ofuscar dichas imágenes para contrarrestarlos (Fumera, Pillai y Roli 2006; Gao, Yan y Choudhary 2009).

1.5.2 Clasificación según el método de diseño

Esta categoría se basa en la necesidad o no de requerir ayuda del usuario para la clasificación de ham y spam o, dicho de otra forma, en necesitar o no la existencia de algún método de aprendizaje automático.

1.5.2.1 Manuales

Fáciles de implementar ya que, a grandes rasgos, se trata de utilizar grandes listas de reglas para filtrar los correos. En dichas listas se distinguen dos tipos, *whitelists* y *blacklists*, las cuales se diferencian en que en las primeras se acepta un correo electrónico si el nombre, la dirección de correo electrónico y la IP del remitente está guardada en éstas (Golbeck y Hendler 2004), mientras que en las *blacklists*, se rechaza dicho correo electrónico si así acaeciera (Ramachandran y Feamster 2006). En esta investigación se aduce que las *blacklists* verán mermada su eficacia en el futuro puesto que han detectado que los *spammers* comienzan a utilizar las IP durante períodos de tiempo muy cortos. Ambos métodos son habitualmente utilizados por los administradores de correos electrónicos y, a mayor escala, también por los remitentes de correos electrónicos. Según (Erickson, Casado y McKeown 2008), tienen un bajo número de falsos positivos y falsos negativos, menores al 1% en ambos casos, pese a que, *a priori*, podríamos pensar que debieran tener menor eficacia por su baja velocidad de adaptación ante cambios, al carecer de sistemas de aprendizaje basados en contenido.

⁴⁴ *Optical Character Recognition* o Reconocimiento Óptico de Caracteres, proceso llevado a cabo a través de la identificación automática de símbolos o caracteres, que pertenecen a un determinado alfabeto, dentro de una imagen. Su uso más frecuente es en la digitalización de textos, desde revistas hasta matrículas de vehículos.

Un método similar pero con mejores perspectivas debido a que hace uso de más información discriminativa es el de *greylists* (Wikipedia 2013d). En éste se combina una *whitelist* con un rechazo temporal de los mensajes cuya tripleta [IP del emisor – dirección de correo del remitente – direcciones de correo de los receptores] no se encuentre dentro de dicha lista. Ello obliga a un reenvío del mensaje por parte del remitente. Presupone que los servidores de correo legítimos lo volverán a intentar enviar mientras que los *spammers* no, debido al coste en tiempo y recursos. Lamentablemente, este método puede ser fácilmente sobrepasado por los *spammers* mediante modificaciones de los paquetes TCP/IP o por ofuscación de la dirección del remitente.

1.5.2.2 Basados en técnicas de aprendizaje automático

Son técnicas que permiten a una máquina aprender por sí misma mediante la generalización de comportamientos a partir del suministro de información no estructurada (ejemplos de entrada). Es decir, la máquina induce conocimientos con mínima o nula ayuda humana. Dicha generalización se realiza a través de la construcción de modelos con los que luego realizar predicciones o tomar decisiones.

Sin lugar a dudas, el más común en la actualidad (Duda, Hart y Stork 2001; Guzella y Caminhas 2009). En este grupo los filtros aprenden en base a unos conjuntos de entrenamiento, para luego comprobar la calidad del diseño y poner en práctica sus conocimientos tanto frente a conjuntos de validación o ajuste como frente a conjuntos de testeo⁴⁵, respectivamente (Keller 2003; Borovicka et al. 2012; Karahoca 2012). De estos subconjuntos (Figura 14), el de ajuste-validación puede tener varios usos puesto que, en el caso de técnicas supervisadas, se emplea para evitar su sobreentrenamiento⁴⁶, mientras que con no supervisadas, esto no se da. En tal caso, este subconjunto sirve principalmente para ajustar y mejorar la optimización de los parámetros de la red.

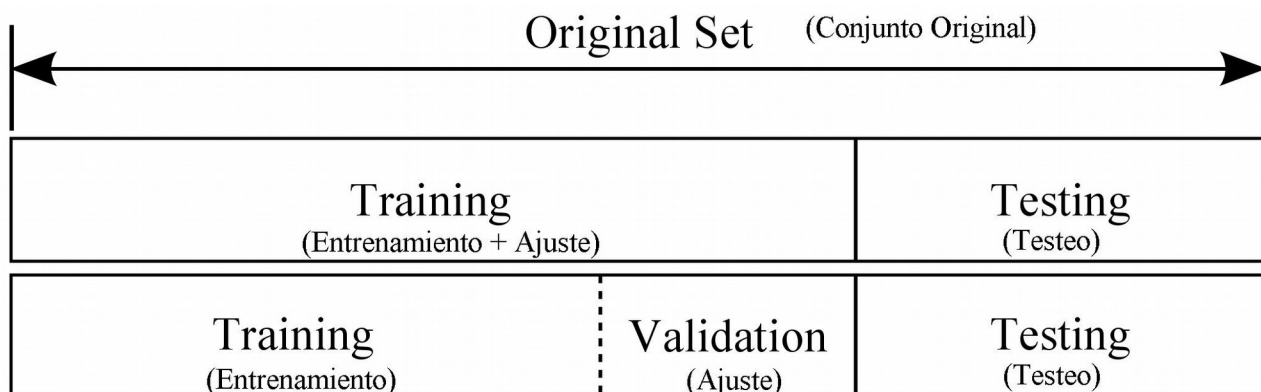


Figura 14: División del conjunto original en tres subconjuntos: entrenamiento, ajuste y testeo.

Posteriormente y en caso de que se utilicen datos simulados, que no es el caso de este proyecto puesto que los datos son reales, se les suelen presentar conjuntos desconocidos más extensos o, directamente, situarlos en entornos “reales” con el fin de que puedan llevar a cabo un proceso de generalización (Álvarez Romero, Domínguez Brito y Suárez Araujo 2013).

A continuación indicamos algunas de las diversas categorías en las que se puede subdividir este gran grupo (Haykin 1999; Duda, Hart y Stork 2001). En nuestro caso, nos hemos basado en la

⁴⁵ Para dichos tres subconjuntos de datos también es común utilizar la nomenclatura en inglés: *training set*, *validation set* y *testing set*, respectivamente.

⁴⁶ También es frecuente emplear su versión inglesa: *overtraining*.

necesidad o no de interacción humana durante el etiquetado de los datos, es decir, en la cantidad de datos etiquetados de guía requeridos durante el aprendizaje. Con ello, en dicha subdivisión obtenemos las siguientes tres agrupaciones: supervisados, no supervisados, y semisupervisados o híbridos. Por último, podemos distinguir dentro de cada una de éstas versiones tanto neuronales (basadas en algún tipo de RNA, como el *SOM*) como no neuronales (usualmente, métodos estadísticos *e.g.* bayesiano).

1.5.2.2.1 Supervisados

Esta clase de filtros se basan en el uso de un conjunto de entrenamiento conformado por pares dato-etiqueta, siendo la etiqueta el valor esperado y deseable que debe devolver el filtro. En el filtrado antispam frecuentemente es correo electrónico-etiqueta, donde la etiqueta puede tener un valor indicador de spam, ham o, en ciertos casos indecisos, "caso gris" (James Bruce 2012). Dicha etiqueta habrá sido definida automática o manualmente, siendo esto último un proceso bastante engorroso y costoso temporalmente. Luego, el filtro deberá clasificar ejemplos nuevos, que carecen de etiqueta.

Dentro de esta clase se pueden distinguir investigaciones que hacen uso de los siguientes clasificadores, siendo el bayesiano el único no basado en una Red Neuronal Artificial:

- Bayesiano, esto es, está basado en el teorema probabilístico de Bayes. Es el más popular, tanto en cantidad de aplicaciones (SpamBayes 2013; The Apache SpamAssassin Project 2013) como en número de investigaciones llevadas a cabo (Sahami et al. 1998; Garcia, Hoepman y Van Nieuwenhuizen 2004; Luo y Zincir-Heywood 2005; Malathi 2011). Por ello, es el más atacado por los *spammers* mediante técnicas como el *Bayesian poisoning*⁴⁷ (Postini 2004; Wittel y Wu 2004; Lowd y Meek 2005; Sprengers y Heskes 2009).
- SVM (Drucker, Wu y Vapnik 1999; Vapnik 1999; Gammerman, Vovk y Vapnik 2013). La SVM intenta construir un hiperplano⁴⁸ con el fin de separar los ejemplares en cierto número de clases dada una función *kernel*, considerándose que esta categorización es mejor cuanto mayor sea el margen⁴⁹. Ha sido objeto de muchas investigaciones, sobre todo desde los años 90, de las que se pueden concluir que la SVM es un buen método, bastante dependiente de la función *kernel* que se use (Chhabra, Wadhvani y Shukla 2010) y cosechando mejores resultados cuando se combinan varios de ellos con una estrategia de votación, reduciéndose los falsos positivos y los errores de clasificación (Blanco, Ricket y Martín-Merino 2007).
- El perceptrón⁵⁰ no suele ser tan común en los filtros antispam actuales (Freund y Schapire 1999; Clark, Koprinska y Poon 2003; Sculley, Wachman y Brodley 2006; Kufandirimbwa y Gatora 2012) como lo es en otros ámbitos como el académico, el reconocimiento de escritura a mano o la detección de bordes. La principal razón de esta menor difusión podría ser, aparte de su simpleza, la preponderancia de los clasificadores bayesianos y SVM.
- LVQ: (Chuan et al. 2005) ha estudiado la versión híbrida de esta RNA, en este caso preprocesando los datos mediante información mutua, explicada en el capítulo 2 (nota al pie 20, página 21). Estos autores afirman que el LVQ cuenta con buenas capacidades de detección de spam (96,20% de *F-measure*, con 98,97% de precisión y 93,58% de sensibilidad) en comparación con otros métodos supervisados, siempre que se realice un

47 Adición de palabras aleatorias o típicas de ham a los mensajes de spam con el fin de degradar la efectividad de los filtros bayesianos al aumentar tanto los falsos negativos como los falsos positivos (Wikipedia 2013b).

48 Extensión del concepto de plano a espacios con más de tres dimensiones. Es un subespacio con una dimensión menos que el espacio donde se encuentra, es decir, separa un espacio en dos semiespacios (Weisstein 2014).

49 Distancia entre el hiperplano y el ejemplar más cercano de cualquiera de las clases.

50 RNA simple que solamente es capaz de resolver problemas linealmente separables. Su versión mejorada, perceptrón multicapa con aprendizaje *Backpropagation*, sí tiene esa capacidad ya que éste sí cuenta con función de activación no lineal.

número suficiente de entrenamientos (al menos 1500 iteraciones).

1.5.2.2.2 No supervisados

También denominado *clustering* (Duda, Hart y Stork 2001) porque categoriza datos en forma de clústers o grupos. En esta clase no existen datos etiquetados de los cuales aprender *ergo* la etapa del preprocesado cobra una mayor importancia. Debido a la mencionada inexistencia de una etapa de etiquetado previo, los datos de entrada suelen:

- Ser más cercanos en el tiempo a los datos reales, al carecer de una etapa de etiquetado previo por parte del usuario (la cual suele ser manual), lo cual es muy positivo dadas las condiciones siempre cambiantes en el entorno de los spam (Cabrera y Acosta 2011).
- Requerir, *a priori*, menos espacio de almacenamiento al no venir acompañados de etiquetas.

Del análisis realizado en la literatura, podemos inferir que no existen tantas investigaciones ni aplicaciones que estén desarrolladas en base a métodos no supervisados. Destacan algunas con muy buenas perspectivas de futuro, demostrando resultados en el mismo orden que los de los métodos supervisados de mayor uso en la actualidad, con la ventaja de no requerir de ejemplos etiquetados. Por su lado, los supervisados han sido más profuda y profusamente investigados, como se indicó, y suelen ser algo más rápidos de entrenar siempre que se hayan obtenido las susodichas etiquetas. Además, en estos el preprocesado no suele tener tanta relevancia como en los primeros.

Algunos sistemas de filtrado no supervisados, de los cuales los tres primeros pertenecen a los no neuronales, son:

- Filtro online *SpamCampaignAssassin (SCA)* (Qian et al. 2010): basado en la existencia de campañas⁵¹ cuyas firmas características las construye mediante *Latent Semantic Analysis (LSA)*⁵².
- Detección de similitudes entre distintas *substrings* dentro de un mensaje (*alienness*): considera spam aquellas que están dentro de una clase equivalente con un alto grado de diferenciación (Narisawa et al. 2007). Las palabras pertenecientes a spam pertenecen a clases equivalentes de similar longitud, tamaño o maximin (nota al pie 20, página 21), las cuales no suelen contener palabras de ham por tener diferentes ocurrencias.
- Uso de árboles de sufijos (*suffix tree*) (Uemura, Ikeda y Arimura 2008): en estos se guardan las relaciones de las ristas de caracteres para estimar la complejidad de un documento y su probabilidad de que sea spam, caracterizados por tener una baja complejidad.
- Uso de Redes Neuronales Artificiales con aprendizaje no supervisado tales como el SOM⁵³:
 - En su versión original (Vrusias y Golledge 2009a, 2009b), investigaciones cuyos preprocesados serán explicados en profundidad en el apartado 2.3.1. En ambas, entrenan al SOM, de 10x10 y durante 1000 ciclos en cada uno de los cuales todas las entradas son presentadas. Este entrenamiento es de manera *online*, es decir, grupos de 100 correos electrónicos son agregados al grupo anterior, recalculados los vectores de características y aprendidos por la red con los pesos obtenidos tras entrenarse con el anterior grupo de correos electrónicos, todo ello repetidamente 1000 veces. Inicialmente, los pesos son aleatorios.
 - Una variante que (Luo y Zincir-Heywood 2005) denominan como SOM Basado en Análisis de Secuencias o *SOM Based Sequence Analysis (SBSA)* en la cual preprocesan la entrada mediante ganancia de información (explicada en el capítulo 2, nota al pie 20,

51 Conjuntos de correos electrónicos de características similares mandados en breves períodos de tiempo.

52 También *Latent Semantic Indexing (LSI)*, asume que palabras con significado similar aparecen en textos parecidos (Landauer y Dumais 2008), construyendo un espacio semántico a partir de textos, pasajes y palabras, y obteniendo similitudes mediante operaciones vectoriales-algebraicas.

53 *Self Organizing Map*, Mapa Autoorganizativo o, más familiarmente, red de Kohonen (Hill 1999).

página 21). Guardan la información temporal y las frecuencias de las palabras en un SOM de doble nivel jerárquico, de 7x13 y 12x12, respectivamente. Finalmente, conectan el segundo SOM a una capa posterior conformada por un k -NN⁵⁴ para la etapa de categorización y análisis de secuencias. Dichos autores han obtenido un alto grado de efectividad, especialmente cuando se consideran muy lesivos los falsos positivos y los falsos negativos.

1.5.2.2.3 Semisupervisados

Se trata de una mezcla entre ambos métodos de aprendizaje anteriormente mencionados que, haciendo uso de unos pocos ejemplares etiquetados (proceso costoso de realizar (Chapelle, Schölkopf y Zien 2006)) y de muchos sin etiquetar (mucho más fáciles de conseguir), consigue buenos resultados.

Dentro de esta clase existen varios métodos semisupervisados, tal y como indican (Chapelle, Schölkopf y Zien 2006; Zhu 2008), los cuales suelen dividirse en:

- Técnicas inductivas: son aquellas en las que el objetivo es la obtención de una función o modelo de predicción definido para todos los casos.
- Métodos transductivos: dícese de aquellos cuya meta es la de obtener predicciones sólo sobre los datos de testeo, sin generalizar ninguna regla de decisión para nuevos casos.
- Combinaciones de ambos, como en (Gao, Yan y Choudhary 2009) con su, según ellos robusto y flexible, *Regularized Discriminant Expectation-Maximization (RDEM)* para la detección de spam basado en imágenes. En éste, la parte transductiva se encarga de etiquetar los datos no etiquetados mientras que, luego, la parte inductiva induce un modelo de mezcla gaussiano con el que poder clasificar nuevos datos. *RDEM* obtiene 91,66% de ratio de detección y menos del 2,96% de ratio de falsos positivos.

Según (Joachims 1999), los métodos transductivos hacen mejores predicciones con menor número de ejemplares etiquetados. Pero presentan un gran hándicap: al no construir un modelo predictivo, la adición de un nuevo ejemplo implica la ejecución del algoritmo transductivo sobre todos los ejemplos de los que predecir etiquetas.

Por otro lado, debido a los buenos resultados que se obtienen en la clasificación de textos con el método supervisado SVM, se ha empezado a investigar una variante semisupervisada de éste: *Transductive Support Vector Machine (TSVM)* (Joachims 1999; Zhou, Burges y Tao 2007; Shunli y Qingshuang 2010) el cual intenta maximizar el margen entre las clases en base a los pocos datos etiquetados que se le han proporcionado. En (Shunli y Qingshuang 2010), *TSVM* presenta mejores resultados que *SVM* mientras que en (Mojdeh y Cormack 2008) son bastante peores debido al uso de conjuntos de entrenamiento anticuados o de distintos usuarios.

Otro equipo ha trabajado con una variante (Pfahringer 2006) del algoritmo *Learning with Local and Global Consistency (LLGC)*, que trata de balancear dos objetivos potencialmente conflictivos: localmente, ejemplos similares deben tener etiquetas de clases similares, y globalmente, las etiquetas predichas deben estar de acuerdo con las de los ejemplos de entrenamiento dados. Sobre ella (Zhou et al. 2004; Santos et al. 2011) aducen que obtienen mejores resultados que con k -NN y SVM. *LLGC* funciona iterativamente, construyendo una *smooth function* (función continuamente diferenciable) en la que cada punto difunde su etiqueta a sus vecinos hasta que se llega a un estado global estable. Esta *smooth function* se construye, principalmente, de

⁵⁴ De *k*-Nearest Neighbor. La clase a la que pertenece un elemento se obtiene observando la de los k elementos vecinos más próximos a éste, siendo la clase ganadora la que más votos tenga.

acuerdo con las siguientes dos suposiciones (Chapelle, Schölkopf y Zien 2006; Zien 2006), las cuales se cumplen en, entre otros, el algoritmo k -NN:

- *Smoothness assumption*: puntos cercanos comparten la misma etiqueta.
- *Cluster assumption*: puntos en la misma estructura o clúster tienen la misma etiqueta.

Por último, la famosa aplicación antispam *SpamAssassin* (The Apache SpamAssassin Project 2013), la cual no se suele clasificar como perteneciente a este grupo sino al supervisado, también puede hacer uso de una función de autoaprendizaje de tipo semisupervisado.

Capítulo 2. Diseño de filtros antispam. Cuerpo de datos y etapa de preprocesado

Introducción: En este capítulo se propone el diseño de filtros antispam usando redes neuronales artificiales no supervisadas. Comenzamos describiendo el cuerpo de datos utilizados en el desarrollo del proyecto. Este conjunto lo constituyen un amplio y representativo grupo de correos electrónicos, procedentes de los diversos cuerpos libres y gratuitos disponibles. El conjunto de datos inicial requiere, para su adecuada preparación, una etapa de preprocesado. Esta etapa está conformada por diversos procesos de limpieza y filtrado que se han diseñado en este proyecto y que también se explican en este capítulo. Por último, detallaremos todas las herramientas software con las que hemos realizado el tratamiento de los datos y todo el desarrollo del proyecto en sí.

2.1 Cuerpo de datos

Una de las bases fundamentales de este proyecto es contar con un conjunto de datos amplio y representativo. Esto significa un cuerpo de datos que contenga correos electrónicos, con una amplia representatividad de spam y ham. La mayoría de repositorios de correo electrónicos suelen ser, desafortunadamente, o bien de pago o bien de acceso restringido, ya sea para mantener la privacidad tanto del emisor como del receptor; o para complicar la labor a los *spammers* al ocultarles las formas de operar de los filtros antispam. En la Tabla 4 se muestran algunos de los repositorios estudiados⁵⁵ y que describiremos seguidamente:

- La base de datos *Sarah's Inbox* (Palin y Sunlight Foundation 2009) se corresponde con unos 14.000 mensajes de la bandeja de entrada de Sarah Palin, antigua gobernadora de Alaska (EEUU)⁵⁶.
- Ling-spam se compone de 481 mensajes de spam recibidos por (Androutsopoulos, Koutsias, Chandrinou, Paliouras, et al. 2000), y 2.412 mensajes de ham provenientes de la lista moderada (*ergo*, libre de spam⁵⁷) Linguist (Linguist 2013).
- Tanto PU1 (con 481 spam y 618 ham) como PU123A (incluye a PU1, además de los conjuntos PU2, PU3 y PUA más recientes y pertenecientes a otros usuarios (Androutsopoulos et al. 2003; Androutsopoulos, Paliouras y Michelakis 2004)) son conjuntos de mensajes encriptados y privados aunque su uso no comercial está permitido. Respecto a las cantidades de ham y spam, PU2 contiene 579 y 142; PU3, 2313 y 1826; y PUA, 571 y 571, respectivamente. Todos estos conjuntos de mensajes fueron limpiados de duplicados, vacíos y, excepto PU3 y PUA, también de los mensajes no escritos en inglés. El spam fue principalmente obtenido del cuerpo de SpamAssassin, explicado más adelante, y de los donados por otros usuarios, aparte de los recibidos por el primero de los autores.
- Spambase es un repositorio de la UCI (*University of California, Irvine*) con más de 4.600 correos electrónicos, de los cuales 1.813 son spam, procesados y etiquetados según el filtro usado por (Hopkins et al. 1999). Este cuerpo de datos lo construyeron en base a ham personales y laborales, y a spam recibidos por su administrador de correo, principalmente.
- Los conjuntos del TREC (*Text REtrieval Conference*; conferencia anual relativa a la Búsqueda y Recuperación de Información) 2005, TREC 2006 y TREC 2007 (National Institute of Standards and Technology 2013; TREC 2005, 2006, 2007) son conjuntos públicos de correos electrónicos, en su mayoría en inglés, ya etiquetados como spam o ham. Entre estos tres conjuntos citados contienen una cantidad superior a los 205.000 mensajes, de los que más de la mitad son spam, tal y como se indica, individualmente, en la Tabla 4.
- El repositorio de SpamAssassin, de ámbito público, está conformado por cerca de 1.900 spam y 4.150 ham. Los mensajes se clasifican en tres grupos: spam, *easy* ham y *hard* ham (estos últimos más proclives a ser identificados erróneamente y considerados como falsos positivos). Fueron recolectados entre 2002 y 2005 (Spam-Assassin 2013; The Apache

55 Todos los cuerpos de correos electrónicos mencionados se caracterizan por ser libres, gratuitos y, exceptuando el primero (ver también nota al pie 56), por haber sido empleados previamente por otros investigadores, principalmente en el desarrollo de filtros antispam o en el de clasificación de mensajes en carpetas.

56 Desafortunadamente, este cuerpo de datos ha dejado de estar disponible ya que fue retirado por el grupo Sunlight Foundation, que lo archivó, analizó y publicó en 2011. Más aún, el dominio de la web enlazada se encuentra en venta actualmente.

57 Cada mensaje, antes de ser publicado en la lista, es moderado por personas, las cuales lo eliminan, por ejemplo, si es spam, si es vacío o irrelevante, o si contiene palabras malsonantes. Por tanto, (Androutsopoulos, Paliouras y Michelakis 2004) suponen que esos moderadores no cometen ningún fallo de clasificación (algo en lo que no está de acuerdo (Graham-Cumming 2006); ver el apartado 1.5) por lo que todos los mensajes que son finalmente publicados pueden considerarse como ham.

SpamAssassin Project 2013).

- La base de datos “Enron Email” (Klimt y Yang 2004; Cohen 2013; Wikipedia 2013b) se compone de más de 619.000 mensajes, de los cuales aproximadamente el 62% son duplicados (Padhye y Pedersen 2006a). Dichos mensajes pertenecían a 158 usuarios, la mayoría de los cuales eran *senior managers*⁵⁸, de la empresa energética estadounidense Enron Corporation. Esta colección de correos se hizo pública a raíz de los problemas financieros y judiciales de esta corporación, los cuales finalmente provocaron su bancarrota en 2001. A su vez, de esta base de datos se han hecho muchos subconjuntos:
 - EnronSent y EnronSubjects (Styler 2011) incluyen 96.106 mensajes y sus líneas de asunto de las carpetas “sent” y “sent-mail”. Estos correos se considerarían ham puesto que son los que fueron enviados manualmente de un empleado de Enron a otra persona, es decir, no son mensajes automáticos. Además, cada uno fue limpiado de firmas automáticas, cabeceras, mensajes anteriores encadenados, etiquetas HTML...
 - Enron-Meetings (Minkov, Wang y Cohen 2005). En este subconjunto, los 733 mensajes fueron extraídos de las carpetas “meetings” y “calendars”, la mayoría sobre reuniones y citas por lo que se podría considerar como un cuerpo de mensajes deseados.
 - Versión preprocesada de Ron Bekkerman (Bekkerman, McCallum y Huang 2004; Bekkerman 2013). Se han eliminado todas las carpetas no temáticas (e.g. bandeja de entrada, enviados, papelera, borradores, etc) de las cuentas de correos de los 7 empleados de Enron que mayor número de mensajes tenían. Además, se han mantenido las carpetas restantes siempre que contuvieran más de dos mensajes (aunque solamente llegando a una profundidad de 2 niveles en la estructura de directorios o árbol de carpetas). En total existen 20.581 mensajes, en 273 carpetas, de los que se desconoce la proporción de mensajes deseados y no deseados puesto que dichas investigaciones versan sobre la clasificación de mensajes en carpetas.
 - El subconjunto de Padhye y Pedersen contiene 3.021 mensajes almacenadas en 6 carpetas temáticas: Negocios, Recursos Humanos (RRHH), Personal, Anuncios generales, EnronOnline⁵⁹ y Mensajes en cadena (Padhye y Pedersen 2006b, 2006b). Trabajaron a partir de la versión preprocesada de (Bekkerman 2013) precedente y, al igual que en ésta, tampoco se proporciona la cantidad de ham y spam por el mismo motivo aducido anteriormente. En este caso, la profundidad de la estructura de directorios ha sido de 5.
 - La versión preprocesada de David Skillicorn (Keila y Skillicorn 2005a, 2005b ; Skillicorn 2013) posee 289.695 correos y existen diversas versiones según el formato de guardado y el preprocesado aplicados. Una de estas versiones se corresponde con dos matrices, donde cada fila es un correo. Las columnas, en una de las matrices son un código numérico que sustituye a una determinada palabra, mientras que en la segunda matriz, sólo existen 4 columnas, cada una correspondiente a la frecuencia de ciertos grupos de palabras asociadas con engaño y fraude. Es por ello que no disponemos tampoco de las cantidades de ham y spam.
- Cuerpo de datos obtenido de la combinación de varios de los conjuntos descritos en párrafos anteriores, además de otros (Metsis, Androutsopoulos y Paliouras 2006a, 2006b). De dicha combinación estos investigadores generaron 6 conjuntos de mensajes con ratios⁶⁰ 1:3 y 3:1

58 Este término inglés puede traducirse de diversas maneras, dependiendo del organigrama, tamaño o sector de la empresa referida: responsable, gerente, directivo, alto ejecutivo, dirigente de alto rango, director, etc. Por ello, en el mercado laboral a nivel internacional suele ser común mantener la denominación en inglés (e.g. CEO o CFO).

59 Plataforma de comercio electrónico para materias primas energéticas creada por Enron en Noviembre de 1999. Dejó de existir en Noviembre de 2002, un año después de haber sido rescatada por el gigante financiero suizo UBS.

60 En matemáticas, la ratio, razón o proporción es una relación binaria entre magnitudes, comparables entre sí, que indica la cantidad de veces que ocurre el primer término (antecedente) con respecto a la del segundo (consecuente). Generalmente se expresa como “a es a b” o “a : b”, aunque en aritmética también se admite la notación como

de ham y spam. La composición de cada uno de estos conjuntos puede verse en la Tabla 5. En este cuerpo los mensajes de ham provienen de la ya mencionada versión preprocesada de (Bekkerman 2013), seleccionando solamente los provenientes de los 6 usuarios de Enron siguientes: “farmer-d”, “kaminski-v”, “kitchen-l”, “williams-w3”, “beck-s” y “lokey-m”⁶¹. Por otro lado, los mensajes de spam los obtuvieron de 4 fuentes distintas (las siglas se utilizan en la Tabla 5 junto a los nombres de usuario anteriores): mensajes no deseados recibidos por Georgios Paliouras (GP), uno de los autores; la colección histórica de spam de Bruce Guenter (BG) (Guenter 1998); el cuerpo de SpamAssassin, ya comentado, y, finalmente, el proyecto *Honey Pot* (Unspam Technologies, Inc 2004). Las siglas SH son utilizadas en aquellos conjuntos de mensajes cuyo spam proviene de estas dos últimas fuentes: SpamAssassin y *Honey Pot*. De este cuerpo de correos electrónicos están disponibles dos versiones:

- Versión sin preprocesar: denominada *raw* (cruda, en bruto, pura o sin editar) en la que sólo se han eliminado los mensajes con virus, los ham enviados a sí mismo por el dueño del correo (es decir, la dirección aparece en los campos “To:”, “CC:” o “BCC:” de la cabecera del correo electrónico), y los spam que usan caracteres no latinos (por ser demasiado fáciles de distinguir como spam).
- Versión preprocesada: basada en esta anterior en la que, además, han sido eliminadas todas las etiquetas HTML y las cabeceras de los mensajes, manteniéndose únicamente el cuerpo y la línea de asunto.

Para nuestro estudio hemos elegido este último cuerpo, combinación de varios, en su versión preprocesada, tanto porque cuenta con mensajes de spam recientes y genéricos como porque es gratuito y de libre acceso. El hecho de que se haya elegido la versión preprocesada del mismo en vez de la versión *raw* revertirá en simplificar la etapa de preprocesado, de suma importancia en el diseño del sistema propuesto. Concretamente, hemos formado un subconjunto de la citada versión combinando únicamente las carpetas “Enron1” y “Enron5” (Tabla 5) con el fin de:

- Balancear el cuerpo de manera que tengamos aproximadamente la misma cantidad (*ergo* ratio 1:1) de ham que de spam: 5172 y 5175, respectivamente.
- Utilizar tanto spam obtenido mediante *honeypots*⁶² y *spamtraps*⁶³ (colecciones SH y BG, sólo utilizada la primera de éstas por estar presente en “Enron 5”) como spam recibido por vías normales (conjunto GP, disponible en “Enron 1”).
- Reducir el número de correos electrónicos con los que trabajar *ergo* simplificar el modelo, acelerar su aprendizaje y reducir el tamaño de los ficheros creados. Aún así, esta reducción mantiene, al mismo tiempo, un conjunto tanto variado y representativo de correos como lo suficientemente amplio para trabajar adecuadamente con el paradigma de la computación neuronal.

cociente a/b Ej.: un ratio 1:3 de ham y spam indica que por cada ham hay 3 spam, es decir, existen 3 veces más cantidad de spam que de ham.

61 Es decir, se ha dejado fuera de este cuerpo, por motivos desconocidos, el conjunto de mensajes del usuario “sanders-r”, que sí está entre los 7 empleados de Enron del cuerpo de (Bekkerman 2013) en el que se basa.

62 Trampa diseñada para atraer, detectar, desviar y contraatacar usuarios malintencionados simulando ser sistemas vulnerables o débiles ante ataques. Pueden ser datos, programas, dispositivos físicos o incluso redes.

63 Tipo especial de *honeypot* dedicado a la captura de correo no deseado. Se basa en la publicación en la Red de direcciones de correo electrónico que están o bien ocultas a los usuarios humanos (Ej.: marcada como no visible o con igual color que el fondo dentro del código HTML) o bien cuya dirección indica claramente que no se debe enviar nada o que no se contesta nunca desde ella (e.g. no-reply@dominio.es, unmanned@domain.com...). Sin embargo, dichas direcciones sí son accesibles para los motores de búsqueda o para los *harvesters* (recolectores de direcciones de correo electrónico) de los *spammers* (Stringhini et al. 2014).

Tabla 4: Repositorios de correos electrónicos públicos, disponibles libre y gratuitamente.

Repositorio	Número de mensajes	Número de spam	Número de ham
Sarah's Inbox	≈ 14.000	N/A ⁶⁴	N/A
Ling-spam (Androutsopoulos, Koutsias, Chandrinos, Paliouras, et al. 2000; Clark, Koprinska y Poon 2003; Schneider 2003; Luo y Zincir-Heywood 2005)	2.893	481	2.412
PU1 (Androutsopoulos, Koutsias, Chandrinos y Spyropoulos 2000; Clark, Koprinska y Poon 2003; Schneider 2003)	1.099	481	618
PU123A (Androutsopoulos, Paliouras y Michelakis 2004)	7.101	4.081	3.020
Spambase (Hopkins et al. 1999)	4.601	1.813	2.788
TREC 2005 (Mojdeh 2012; Chhabra, Wadhvani y Shukla 2010; Mojdeh y Cormack 2008)	92.189	52.790	39.399
TREC 2006 (Cormack y Mojdeh 2009)	37.822	24.912	12.910
TREC 2007 (Mojdeh 2012; Mojdeh y Cormack 2008)	75.419	50.199	25.220
SpamAssassin (Pitsillidis et al. 2010; Chuan et al. 2005; Khorsi 2007; Garcia, Hoepman y Van Nieuwenhuizen 2004; Anderson 2006; Mojdeh y Cormack 2008)	6.047	1.897	4.150
Enron Email (Klimt y Yang 2004; Padhye y Pedersen 2006a, 2006b ; Pitsillidis et al. 2010; Chhabra, Wadhvani y Shukla 2010; Mojdeh 2012; Cohen 2013)	619.446	N/A	N/A
Combinación de varios (Metsis, Androutsopoulos y Paliouras 2006a, 2006b)	33.716	16.545	17.171

Tabla 5: Conjunto de datos seleccionado. Se han destacado en **negrita** los dos subconjuntos de correos electrónicos que conformarán el de nuestra investigación, nuestro cuerpo de datos.

Nombre de la carpeta / conjunto de mensajes	Origen (ham + spam)	Cantidad (ham:spam)	Períodos ham	Períodos spam
Enron1	farmer-d + GP	3672:1500	12/1999 - 01/2002	12/2003 - 09/2005
Enron2	kaminski-v + SH	4361:1496	12/1999 - 05/2001	05/2001 - 07/2005
Enron3	kitchen-l + BG	4012:1500	02/2001 - 02/2002	08/2004 - 07/2005
Enron4	williams-w3 + GP	1500:4500	04/2001 - 02/2002	12/2003 - 09/2005
Enron5	beck-s + SH	1500:3675	01/2000 - 05/2001	05/2001 - 07/2005
Enron6	lokay-m + BG	1500:4500	06/2000 - 03/2002	08/2004 - 07/2005

2.2 Sistema inteligente para la detección de spam

⁶⁴ N/A: en la Tabla 4 tendrá el significado de *Not Available*, es decir, No Disponible.

El sistema de filtrado antispam que hemos propuesto en este trabajo consta de dos etapas bien diferenciadas y que son aplicadas en el orden expuesto en la Figura 15:

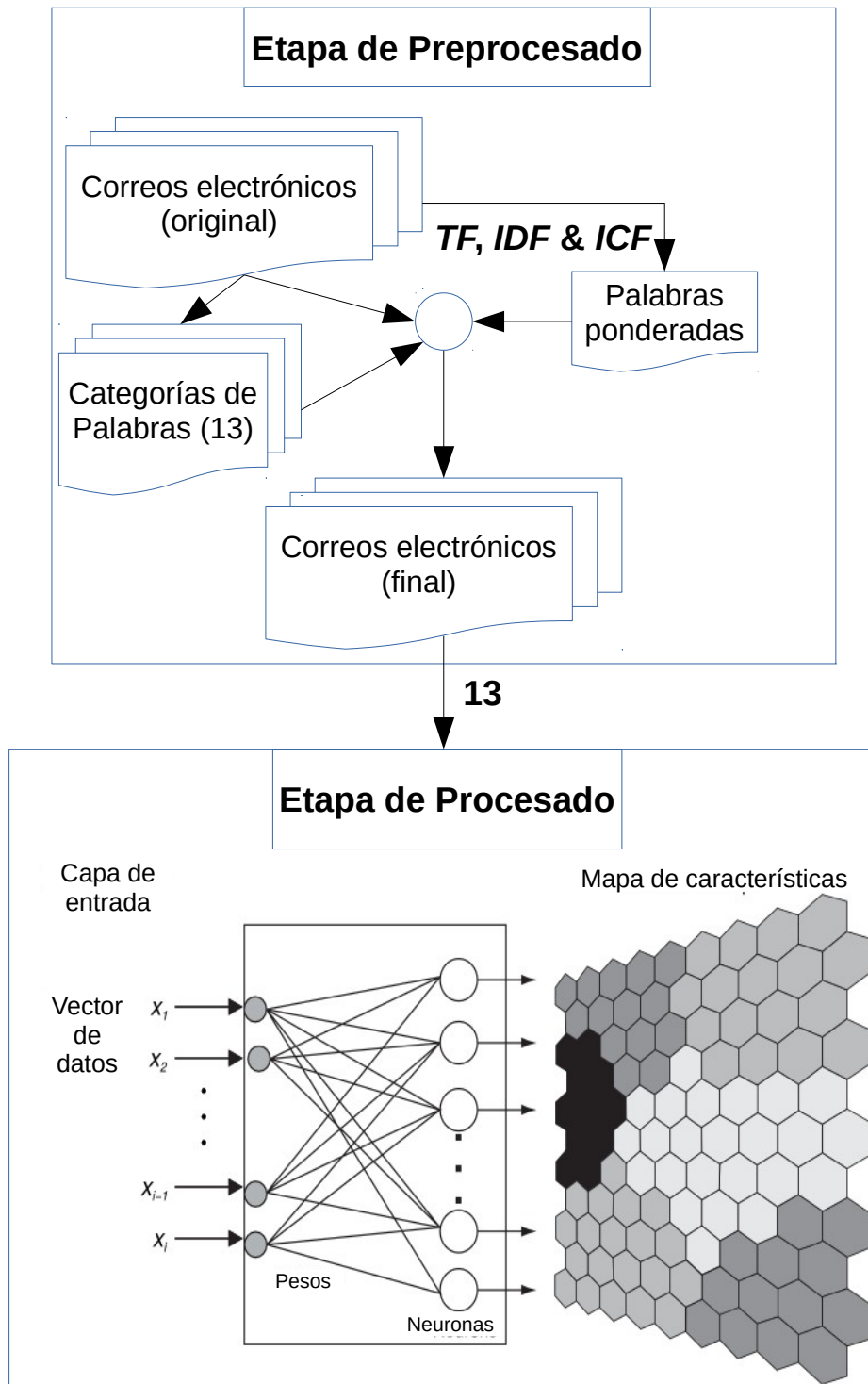


Figura 15: Esquema del sistema inteligente para la detección de spam.

1. Etapa de preprocesado: la hemos construido en base, por un lado, a las métricas *TF*, *IDF* e *ICF* y, por otro lado, a la existencia de categorías temáticas en los spam. Tanto esta etapa en sí como las anteriores métricas serán explicadas más profundamente en el apartado 2.2.1. La importancia de esta etapa radica en que acelera y facilita la consecución de la siguiente.

2. Etapa de procesado: basada en los mencionados mapas autoorganizados, redes de Kohonen o SOM. Expondremos el diseño de esta etapa en el apartado 2.2.2 (escuetamente ya que será explicado y justificado en profundidad al comienzo del Capítulo 3), cuyos resultados serán presentados en el siguiente capítulo.

2.2.1 Etapa de preprocesado

El preprocesado es una etapa muy importante (Hovold 2005; Kotsiantis, Kanellopoulos y Pintelas 2006; Guzella y Caminhas 2009; Zhang 2012) – e, incluso, crítica (Álvarez Romero, Domínguez Brito y Suárez Araujo 2013) – para el aprendizaje de cualquier tipo de RNA, especialmente las relacionadas con el aprendizaje no supervisado. Ello es debido a la necesidad de obtener información tanto de mayor calidad como, usualmente, en menor cantidad que la original, que suele ser caracterizada como incompleta, ruidosa e inconsistente. Algunas de las técnicas utilizadas para disponer de información con las dos características indicadas son:

- Para que sea de mayor calidad: limpieza de la información, selección o extracción de las características más relevantes, normalización de los datos, transformación de los datos, etc.
- Para que sea en menor cantidad: reducción de dimensiones, discretización de los datos...

Comenzaremos explicando el preprocesado aplicado en otras investigaciones para, posteriormente, hacer lo propio con el desarrollado en este proyecto.

2.2.1.1 Otros autores

Actualmente, las investigaciones sobre filtros antispam han utilizado diferentes métodos de preprocesamiento de sus respectivos cuerpos de correos electrónicos. Debemos destacar, debido a su relación con nuestro preprocesado, dos trabajos (Vrusias y Golledge 2009a, 2009b), ambos basados en el producto de las siguientes dos métricas, en sus formas normalizadas:

- *Weirdness*: indica la rareza del token⁶⁵. Compara la frecuencia de cierto token en el dominio de spam frente a la que tiene en el *BNC*⁶⁶, teniendo un mayor valor cuanto menos frecuente es ese término en el lenguaje común.
- *TF*IDF*⁶⁷: es mayor cuanto más frecuente es un término en un documento particular pero infrecuente considerando todo el conjunto de documentos.

El producto de ambas métricas indica el grado de *spaminess*⁶⁸, R en la Fórmula 7, de cierto token t .

$$R_t = weirdness_t \times tf*idf_t \quad (7)$$

Por una parte, el valor de *weirdness* del token t de spam, véase la Fórmula 8, se calcula como el cociente de, por un lado, la división de la frecuencia f_s de dicho token con el número total de frecuencias de tokens N_s (en el conjunto de spam) y, por otro lado, de la división de la frecuencia

65 En el presente documento, el término "token" será sinónimo de "palabra", a no ser que se indique otra cosa.

66 *British National Corpus* (Brigham Young University 1994). Es una colección de algo más de 1 millón de palabras inglesas de diferentes fuentes, orales y escritas, que sirve de muestra de dicho idioma.

67 *Term Frequency * Inverse Document Frequency* o Frecuencia del Término * Inversa de la Frecuencia del Documento.

68 Indica la probabilidad relativa de que algo (token, correo electrónico, comentario...) sea spam. Antónimo de *haminess* o cuán probable es que esté relacionado o sea considerado como ham.

f_{BNC} del token con el número total de frecuencias de tokens N_{BNC} (en el conjunto genérico del *BNC*).

$$weirdness_{t_s} = \frac{f_{t_s} / N_{t_s}}{f_{t_{BNC}} / N_{t_{BNC}}} \quad (8)$$

Por otra parte, el valor de la $TF*IDF$ del token t de spam, tal y como viene indicado en la Fórmula 9, es obtenido mediante el cociente de la frecuencia de dicho término f_s (TF) y de la suma de frecuencias de cada token en el conjunto de spam y ham $f_{s,H}$ (IDF).

$$tf * idf_{t_s} = \frac{f_{t_s} \times \log_2\left(\frac{N}{n_t}\right)}{\sqrt{\sum_1^{t_{s,H}} \left(f_{t_{s,H}} \times \log_2\left(\frac{N}{n_t}\right)\right)^2}} \quad (9)$$

La principal diferencia entre ambos trabajos de los mismos autores es que (Vrusias y Golledge 2009a) utiliza el top 500 de tokens de spam con lo que el vector, binario, de entrada tiene 501 atributos: 500 para spam más uno adicional para indicar la presencia de tokens de ham. En cambio, (Vrusias y Golledge 2009b) usa muchos menos (26 atributos: 25 para spam además de uno con la misma finalidad que en el trabajo anterior) para reducir el tiempo de aprendizaje y el consumo de recursos. Cada uno de esos 25 atributos está conformado por una agrupación de tokens de spam, en cantidad variable dentro de cada agrupación, ordenados según un ranking de *spaminess* (R en la Fórmula 7). Es común la presencia de tokens con bajo ranking tanto en spam como en ham. Cada uno de esos atributos indica la proporción – por tanto, ahora el vector ya no es binario sino con valores comprendidos entre 0 y 1 – de tokens dentro de esa agrupación presentes en el correo electrónico. Dicho ranking de *spaminess* viene dado por el producto del *weirdness* con la $TF-IDF$, es decir, por R según la Fórmula 7.

Por otra parte y relativos al uso de categorías o clases, son de gran interés tres trabajos relacionados con la $TF*ICF$ ⁶⁹ para la clasificación de textos (Lertnattee y Theeramunkong 2004; Rong Jin, Joyce Y. Chai y Luo Si 2005; Wang y Zhang 2013). Esta relevancia para nuestro proyecto se debe a la existencia de categorías de palabras por lo que tiene mayor sentido utilizar la ICF y no la IDF para la atenuación⁷⁰ de las TF de las palabras en el cálculo de sus pesos. La razón de ello es que la ICF es relativa al número de categorías en las que aparece una palabra y no al de documentos, como en el caso del IDF . La utilización de la ICF se basa en la siguiente suposición (Wang y Zhang 2013): "*a menor cantidad de categorías en las que aparece cierto token, mayor poder de discriminación tendrá éste en la categorización de textos*", es decir, la ICF favorece los términos raros atenuando los más comunes a nivel de categoría o clase, en vez de a nivel de documentos como ocurre con IDF .

Uno de estos trabajos, (Lertnattee y Theeramunkong 2004), propone la combinación de IDF e ICF en la forma $TF*IDF*ICF$. Además, define tres funciones diferentes para calcular la ICF (véanse las Fórmulas 10 a 12).

69 *Term Frequency * Inverse Category Frequency* o Frecuencia del Término * Inversa de la Frecuencia de la Categoría. También, la C en la ICF puede ser relativa a una clase en vez de a una categoría.

70 Entiéndase "atenuación de la frecuencia de un término" como "reducción de su poder de discriminación".

$$ICF_{Loga} = \log\left(\frac{|C|}{f_t}\right) \quad (10)$$

$$ICF_{Linear} = \frac{|C|}{f_t} \quad (11)$$

$$ICF_{Sqr} = \sqrt{\frac{|C|}{f_t}} \quad (12)$$

donde C es el número total de categorías o clases, y f_t , la frecuencia o cantidad de veces que aparece el token t en cierta categoría o clase.

Por otro lado, en la Tabla 6 y Tabla 7 podemos observar, respectivamente, los métodos de selección de características, y las dimensiones de los vectores de entrada a las redes, utilizados en otras investigaciones. Más aún, de los trabajos indicados en la Tabla 7 podemos extraer como conclusión que una mayor cantidad de atributos implica mejor clasificación en RNA Supervisadas – mejora despreciable dado el mayor coste computacional según (Metsis, Androutsopoulos y Paliouras 2006a) – no cumpliéndose necesariamente con RNA No Supervisadas, más dependientes de la calidad de los propios atributos que de su cantidad.

Tabla 6: Métodos de selección de características utilizados por otros investigadores.

Método de selección de características	Investigadores
IG^{71} o MI^{72}	(Drucker, Wu y Vapnik 1999; Androutsopoulos, Paliouras, Karkaletsis, Sakkis, et al. 2000; Clark, Koprinska y Poon 2003; Schneider 2003; Kolcz, Chowdhury y Alspector 2004; Luo y Zincir-Heywood 2005; Hovold 2005; Chuan et al. 2005)
$TF*IDF$	(Clark, Koprinska y Poon 2003; Vrusias y Golledge 2009b, 2009a ; Qian et al. 2010)
n-gramas de palabras ⁷³	(Hovold 2005)
Rareza o <i>alienness</i> ⁷⁴	(Narisawa et al. 2007)
Varianza ⁷⁵	(Clark, Koprinska y Poon 2003)
DCE^{76}	(Uemura, Ikeda y Arimura 2008)

71 *Information Gain* o Ganancia de Información; propiedad estadística que mide cómo clasifica ese atributo a los ejemplos. Se calcula determinando la diferencia entre la entropía del conjunto de datos de partida y la suma ponderada de las entropías una vez dividido el conjunto de ejemplos:

$$Gain(S, A) = Entropia(S) - \sum_{v \in \text{valores}(A)} \frac{|S_v|}{|S|} Entropia(S_v)$$

72 *Mutual Information*, Información Mutua o transinformación; mide la reducción de la incertidumbre de una variable debido al conocimiento de otra, ambas aleatorias. Se puede calcular como la diferencia entre la entropía de la clase A y la entropía condicional de A (respecto del atributo B):

$$I(A; B) = H(A) - H(A|B) = H(B) - H(B|A) = I(B; A) = H(A) + H(B) - H(A, B)$$

73 Subsecuencia de n elementos (fonemas, letras, palabras, etc) de una secuencia escrita u oral dada.

74 Indica cuán diferente es un mensaje con respecto a otro en términos de longitud, tamaño y *maximin* (diferencia

Tabla 7: Dimensiones de los vectores de entrada a la red utilizadas en otros trabajos.

Dimensión de los vectores de entrada	Trabajos
26	(Vrusias y Golledge 2009b)
50-700, con pasos de 50	(Androutsopoulos, Paliouras, Karkaletsis, Sakkis, et al. 2000)
50-750, con pasos de 100	(Luo y Zincir-Heywood 2005)
100	(Chuan et al. 2005)
256	(Clark, Koprinska y Poon 2003)
501	(Vrusias y Golledge 2009a)
50-5.000, con pasos de 50	(Schneider 2003)
500, 1.000 ó 3.000	(Metsis, Androutsopoulos y Paliouras 2006a)
500-4.000	(Hovold 2005)
1000, +6.570 ó +7.450	(Drucker, Wu y Vapnik 1999)

2.2.1.2 Preprocesado híbrido

El preprocesado que hemos desarrollado es híbrido puesto que combina una etapa manual con otra automática. Seguidamente expondremos los objetivos de nuestro preprocesado y explicaremos las características del mismo, incluyendo las tres fases que conforman ambas etapas.

2.2.1.2.1 Objetivos

Los objetivos de nuestro preprocesado son idénticos al de otros preprocesados (Kotsiantis, Kanellopoulos y Pintelas 2006; Zhang 2012):

- Reducir la dimensionalidad del vocabulario, es decir, disponer de menor número de palabras con el fin de facilitar y acelerar el aprendizaje de la RNA.
- Ponderar las palabras de manera que sólo utilicemos las que tengan mayor importancia⁷⁷, desechando tanto las palabras más comunes (*stop words*) como las que aparezcan muy pocas veces.

2.2.1.2.2 Categorías de palabras

Nuestro preprocesado está basado en la premisa de que en los spam existen diversas categorías temáticas (Wang, Irani y Pu 2013; Trustwave 2013a, 2013b ; Malathi 2011; Vrusias y

entre la longitud de un representante y la del elemento mínimo más largo de la clase equivalente, indicando cuánto se puede acortar el representante y que siga perteneciendo a la misma clase equivalente).

75 Mide la dispersión de un conjunto de datos y se calcula como la esperanza o valor esperado del cuadrado de la diferencia entre dicha variable y su media aritmética o promedio:

$$\sigma^2 = \frac{\sum (X - \mu)^2}{N}$$

76 *Document Complexity Estimation* o Estimación de la Complejidad del Documento; algoritmo probabilístico que calcula la similitud – tras producirse posibles mutaciones (cambios aleatorios de letras en un documento), *crossover* (intercambios aleatorios de partes entre varios documentos) y ensalada de palabras (reemplazar un conjunto aleatorio de palabras de una oración con los de otra obteniendo una nueva gramaticalmente correcta pero carente de sentido) – entre un documento y cierta *spam seed* (semilla de spam) de manera que parezcan copias imperfectas.

77 Entiéndase como "las palabras con mayor poder discriminador".

Golledge 2009b, 2009a ; Chuan et al. 2005; Chhabra 2005; Sahami et al. 1998), *ergo*, en base a ello sería factible su detección y diferenciación con respecto al ham.

Tal y como podemos observar en la Tabla 8, hemos seleccionado 13 categorías de spam o ham presentes en el cuerpo elegido. Estas categorías también pueden ser, por un lado, clasificadas visualmente, tipo semáforo, según su grado de *spaminess*, y, por otro lado, extrapolables a cualquier otro cuerpo tal y como se deduce de la Tabla 9.

El número de palabras dentro de cada categoría, como es de esperar, no es idéntico, cantidades que varían entre 20 y 4.500. Con este preprocesado conseguimos reducir la dimensionalidad en varios órdenes de magnitud, consiguiéndose que la selección del cuerpo analizado – conjuntos de correos “Enron1” y “Enron5” completos tal y como se indicó en la Tabla 5 – solamente contenga 66.650 palabras diferentes, de tamaño mayor a 2 caracteres.

Tabla 8: Clasificación visual tipo semáforo de las 13 categorías de palabras según su grado de *spaminess*.

Color semáforo	Significado	Categorías de palabras
ROJO	Mayor <i>spaminess</i> & Menor <i>haminess</i>	Sexo/Relaciones
		Medicina/Drogas/Cuerpo
		Apuestas/Azar/Adivinación
		Banca/Inversiones/Seguros/Compra-Venta
		Enlace/Correo electrónico
		Otros idiomas
		Ofuscación (mal escritos a propósito)
ÁMBAR	Casos grises (James Bruce 2012)	Negocios/Empresa/Gobierno
		Internet/Tecnología
VERDE	Menor <i>spaminess</i> & Mayor <i>haminess</i>	Documentos/Trámites
		Nombres/Familia
		Turismo/Regiones
		Fichero adjunto

Cabe mencionar con mayor detalle algunas de las categorías de palabras anteriores:

- “Ofuscación”⁷⁸: también bastante, sino la más, indicadora de spam ya que los *spammers* intentan ocultar las palabras con el fin de dificultar la labor a los filtros antispam, especialmente a los basados en Bayes (Wikipedia 2013a), con el riesgo, y obvia desventaja, de que el mensaje sea poco o nada comprensible para el usuario final.
- “Adjuntos”: indica la presencia de ficheros adjuntos en el correo electrónico, solamente común a los ham en este cuerpo.
- “Otros idiomas”: debido a que el cuerpo de correos electrónicos es en inglés, la presencia de palabras en otros idiomas – español, francés, alemán y holandés, principalmente – puede ser

⁷⁸ Palabras escritas incorrectamente a propósito mediante, entre otros métodos, desorden, troceado, mezcla, código HTML, y uso de otros caracteres de apariencia similar (Ej.: como algunos pertenecientes al alfabeto cirílico que visualmente son idénticos a otros del alfabeto latino pero usan una codificación distinta) (Freschi, Seraghiti y Bogliolo 2006; Liu y Stamm 2007; Sprengers y Heskes 2009)

un buen indicador de spam.

Es importante indicar que las palabras, que pueden pertenecer a varias categorías al mismo tiempo⁷⁹ (Ej.: “Ofuscación” y “Medicina”, “Sexo” y “Enlace”, etc), se clasificaron manualmente, proceso que no tendría que volverse a hacer nuevamente, ni por los usuarios ni por los desarrolladores. La principal ventaja de que se haya hecho a mano es, pese al coste temporal de hacerlo, la robustez frente a los intentos de ofuscación de las palabras, de ahí la existencia, como se había mencionado, de una categoría que representa dicha contingencia. Esta ofuscación es comúnmente utilizada por los *spammers*, como se demostrará en las Tabla 11 y Tabla 13. Otra ventaja es que no se requiere aplicar ningún proceso de lematización⁸⁰, bastante común en el resto de trabajos consultados aplicándose más concretamente la variante implementada por (Porter 1980).

79 Es decir, las categorías de palabras no son exclusivas: existe solapamiento entre ellas. La bondad de esta manera de proceder es que, cuando existe solapamiento, ayuda a reafirmar el carácter spam o ham de cierta palabra.

80 Obtención del lema o raíz de la palabra. Ej.: *fucking* → *fuck*; *bitches* → *bitch*;

Tabla 9: Categorías de spam existentes según diversos autores (destacado en **negrita el nuestro**).

Autores → Categorías ↓	<i>Trustwave (Spam)</i>	<i>Malathi</i>	<i>Vrusias</i>	<i>Chuan</i>	<i>Chhabra</i>	<i>Sahami</i>	Ylermi
<i>Porn / Sexo</i>	X	X	X	X	X	X	X
<i>Dating / Relaciones</i>	X		n/a	n/a		n/a	
<i>Pharmaceuticals / Medicina</i>	X	X	n/a	n/a	X	n/a	X
<i>Phishing</i>	X	X	n/a	n/a	X	n/a	n/a
<i>Scam / Azar</i>	X	X	n/a	n/a	n/a	n/a	X
<i>Gambling / Casinos</i>	X	X	n/a	n/a	n/a	n/a	
<i>Adivinación</i>	n/a	n/a	n/a	n/a	n/a	n/a	
<i>Financial / Banca</i>	X	n/a	n/a	X	X	n/a	X
<i>Stock / Inversiones</i>	X	X	n/a			n/a	
<i>Replicas / Compra-venta</i>	X	X	n/a	X	X	n/a	
<i>Seguros</i>	n/a	n/a	n/a	n/a		n/a	
<i>Diplomas</i>	X	X	n/a	n/a	X	n/a	n/a
<i>Vacaciones</i>	n/a	n/a	n/a	X	X	n/a	n/a
<i>Software / IT</i>	X	X	X	X	X	n/a	X
<i>Malware</i>	X	n/a	n/a	n/a	n/a	n/a	n/a
<i>Malware links</i>	X	n/a	n/a	n/a	n/a	n/a	n/a
<i>Mule jobs / Trabajos falsos</i>	n/a	X	n/a	n/a	n/a	n/a	n/a
<i>Política</i>	n/a	n/a	n/a	X	n/a	n/a	n/a
<i>Correos electrónicos / Enlaces</i>	n/a	n/a	n/a	n/a	n/a	n/a	X
<i>Ofuscación</i>	n/a	n/a	n/a	n/a	n/a	n/a	X
<i>Otros idiomas</i>	n/a	n/a	n/a	n/a	n/a	n/a	X
<i>Otros spam</i>	X	n/a	n/a	X	X	X	n/a

En cuanto a los porcentajes de ham y spam dentro de cada categoría, los cálculos sobre la totalidad de nuestro cuerpo arrojan los resultados de la Tabla 10 y Tabla 11. Llevando a cabo idénticas operaciones pero en esta ocasión sobre parte del cuerpo – más concretamente sobre el 80%, relativo al conjunto de entrenamiento y validación o ajuste, y dejando fuera al 20% restante, correspondiente al conjunto de testeo – se obtuvieron los resultados de la Tabla 12 y Tabla 13.

Tabla 10: Porcentajes de ham y spam de las categorías 1-6 en todo nuestro cuerpo.

Categorías Porcentajes	Sexo	Medicina	Azar	Compras	Enlaces	Otros idiomas
Ham	0,19%	0,44%	0,09%	2,8%	1,04%	0,96%
Spam	5,43%	5,35%	2,39%	8,05%	3,91%	1,01%

Tabla 11: Porcentajes de ham y spam de las categorías 7-13 en todo nuestro cuerpo.

Categorías Porcentajes	Ofuscación	Negocios	Tecnología	Documentos	Nombres	Lugares	Adjuntos
Ham	0,04%	18,29%	1,08%	7,27%	2,14%	1,01%	32,92%
Spam	2,58%	0,61%	9,48%	0,36%	0,19%	1,69%	0,55%

Tabla 12: Porcentajes de ham y spam de las categorías 1-6 en el conjunto entrenamiento-ajuste.

Categorías Porcentajes	Sexo	Medicina	Azar	Compras	Enlaces	Otros idiomas
Ham	0,2228%	0,3939%	0%	2,5508%	0,9584%	0,9070%
Spam	4,9968%	4,7855%	2,4096%	7,2939%	3,4197%	0,9598%

Tabla 13: Porcentajes de ham y spam de las categorías 7-13 en el conjunto entrenamiento-ajuste.

Categorías Porcentajes	Ofuscación	Negocios	Tecnología	Documentos	Nombres	Lugares	Adjuntos
Ham	0,0393%	15,8043%	0,8776%	6,7625%	1,8509%	1,0672%	31,3982%
Spam	2,2772%	0,5405%	8,5757%	0,2940%	0,2057%	1,7076%	0,6541%

Como se puede observar en las Tabla 10 a Tabla 13, tanto para parte del cuerpo como para su totalidad las categorías "Ofuscación", "Sexo", "Azar" y "Medicina" son mucho más comunes – aproximadamente entre 1 y 2 órdenes de magnitud más – en spam que en ham mientras que ocurre justamente lo contrario con las categorías "Documentos", "Adjuntos", "Negocios" y "Nombres", similar a lo que se había inferido anteriormente mediante la clasificación visual tipo semáforo.

2.2.1.2.3 Eficiencia en la implementación

Con el fin de acelerar el preprocesado pero sin requerir un uso excesivo de espacio de almacenamiento, las palabras de todas las 13 categorías se almacenaron en varios ficheros de texto, tantos como categorías hay, cuyos contenidos son accedidos a través de igual cantidad de tablas de hash⁸¹, contenedores específicamente preparados para almacenar parejas clave-valor y, al mismo tiempo, disponer de accesos eficientes (Roca et al. 2005): búsquedas, inserciones y eliminaciones en orden $O(1)$, tanto en el mejor caso como el caso promedio, y de $O(n)$ en el peor caso.

Por otra parte, también ayudaron a mejorar el rendimiento de la etapa de preprocesado de los

81 También denominadas, según el lenguaje de programación o entorno de desarrollo utilizado, como *container maps* (MATLAB), *maps* (STL de C++), diccionarios (Python), correspondencias o *arrays* asociativos.

datos el uso de la *Parallel Computing Toolbox* para la ejecución en paralelo junto al análisis del código y aplicación de técnicas eficientes (Ej.: preasignación del tamaño final de vectores y *cell arrays*, declaración de variables dentro del bloque donde se usarán, cierre de ficheros en desuso, etc), muchas de ellas aprendidas durante los estudios.

2.2.1.2.4 Fases del preprocesado

Nuestro preprocesado se compone de tres fases, explicadas con mayor detalle en los subapartados siguientes, cuya ejecución debe ser en el orden indicado en la Figura 16 ya que son interdependientes:

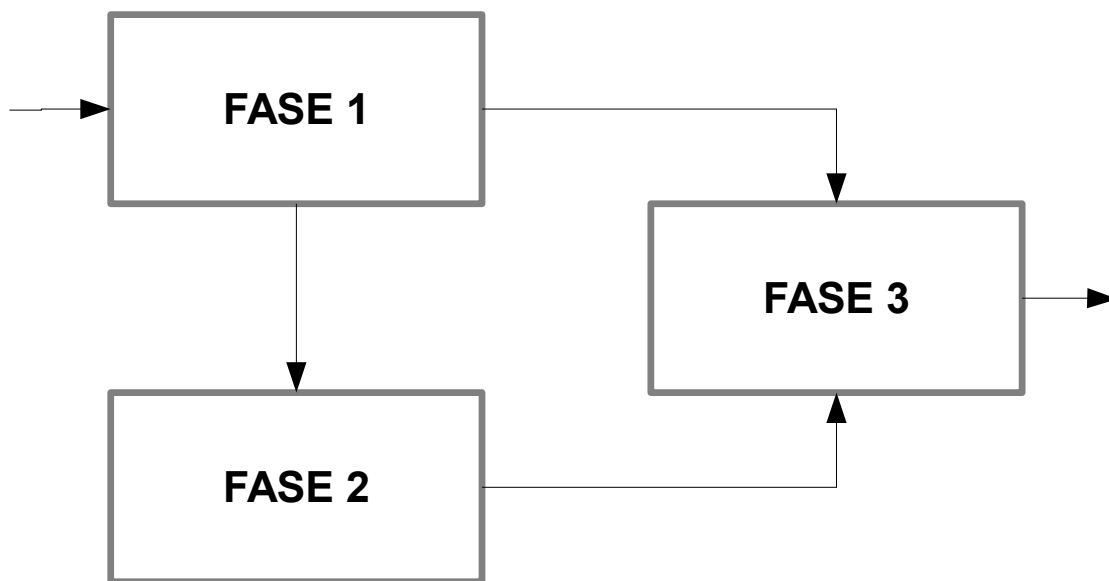


Figura 16: Diagrama de flujo sobre las relaciones entre las distintas fases de nuestro preprocesado.

2.2.1.2.4.1 Fase 1 del preprocesado

Inicialmente, se realiza el particionado del cuerpo en tres subconjuntos de datos (Keller 2003; Borovicka et al. 2012; Karahoca 2012), tal y como se explicó en la Figura 14:

- Entrenamiento: 60% del cuerpo. Se utilizará para entrenar cada una de las variantes de RNA que se decidan comprobar.
- Ajuste o validación: 20% del cuerpo. Servirá para comprobar la calidad de las diversas variantes de RNA que se han creado, afinándolas y quedándonos con las mejores opciones (conjunto de soluciones finales).
- Testeo: 20% del cuerpo. Confirmará la calidad del conjunto de soluciones finales, comprobándolo sobre datos que no han sido vistos anteriormente.

Seguidamente y sobre la unión entre los conjuntos de datos para el entrenamiento con el que es para el ajuste⁸², se obtienen las *raw TF*⁸³ de ciertas palabras, a las que hemos denominado “selectas” debido a que deben reunir ciertas características:

82 Siguiendo el principio de Pareto (Wikipedia 2014), se unificaron temporalmente ambos conjuntos en uno solo, con el 80% del cuerpo y formado por datos ya vistos, sobre el que se aplicarán todas las fases de nuestro preprocesado.

83 *Raw Term Frequency* o Frecuencia Bruta del Término. Señala la multiplicidad o número de veces que ocurre un determinado término en un documento concreto, sin aplicarse ningún método de normalización. Es considerada la versión más sencilla de implementar así como la más rápida de ejecutar de todas las posibles *TF*.

- Pertenecer a la línea de asunto o al cuerpo del correo electrónico⁸⁴.
- Sólo contener letras, sin números ni signos de puntuación.
- No ser comunes, es decir, no estar incluidas en un fichero de *stop words*⁸⁵, en nuestro caso obtenido de la herramienta *TMG* (Zeimpekis, Kontopoulou y Gallopoulos 2011).

La salida de esta fase, similar al preprocesado de (Skillicorn 2013), es un archivo de texto donde cada una de las líneas, que están en formato *CSV*⁸⁶ y representan un único correo electrónico, están conformadas por pares de palabras *selectas* y sus respectivas *raw TF*. Además, dichas palabras están ordenadas alfabéticamente por lo que se ha perdido el orden original de las palabras, algo común en implementaciones del modelo *bag-of-words*⁸⁷. Debemos mencionar que no existe ninguna distinción entre si una determinada palabra se encontraba en el cuerpo o en el asunto de dicho correo electrónico, es decir, son tratadas igual. Al final de cada línea se le ha añadido dos etiquetas que indican ham o spam, y el nombre del fichero tratado, tal y como se puede ver en la Figura 17.

84 Los correos electrónicos fueron preprocesados previamente – en la que podríamos llamar como la fase 0 de nuestro preprocesado – por (Metsis, Androutsopoulos y Paliouras 2006a), proceso que se basó en, entre otras cosas, seleccionar sólo el cuerpo y el asunto, y colocar los signos de puntuación entre dos espacios. En el cuerpo original, cada uno de esos correos electrónicos está guardado en su propio fichero de texto, y dentro de un determinado directorio.

85 Palabras que son demasiado frecuentes en el lenguaje y, por tanto, comunes tanto a spam como a ham, tales como preposiciones (*at, in, on*), conjunciones (*and, or*), artículos (*a, the*), pronombres (*our, you*), etc.

86 *Comma-Separated Values* o Valores Separados por Comas (Shafranovich 2005).

87 El modelo de la "bolsa de palabras" es una representación simplificada de un texto (frase o documento) obviando la gramática y el orden de las palabras pero manteniendo la multiplicidad de éstas.

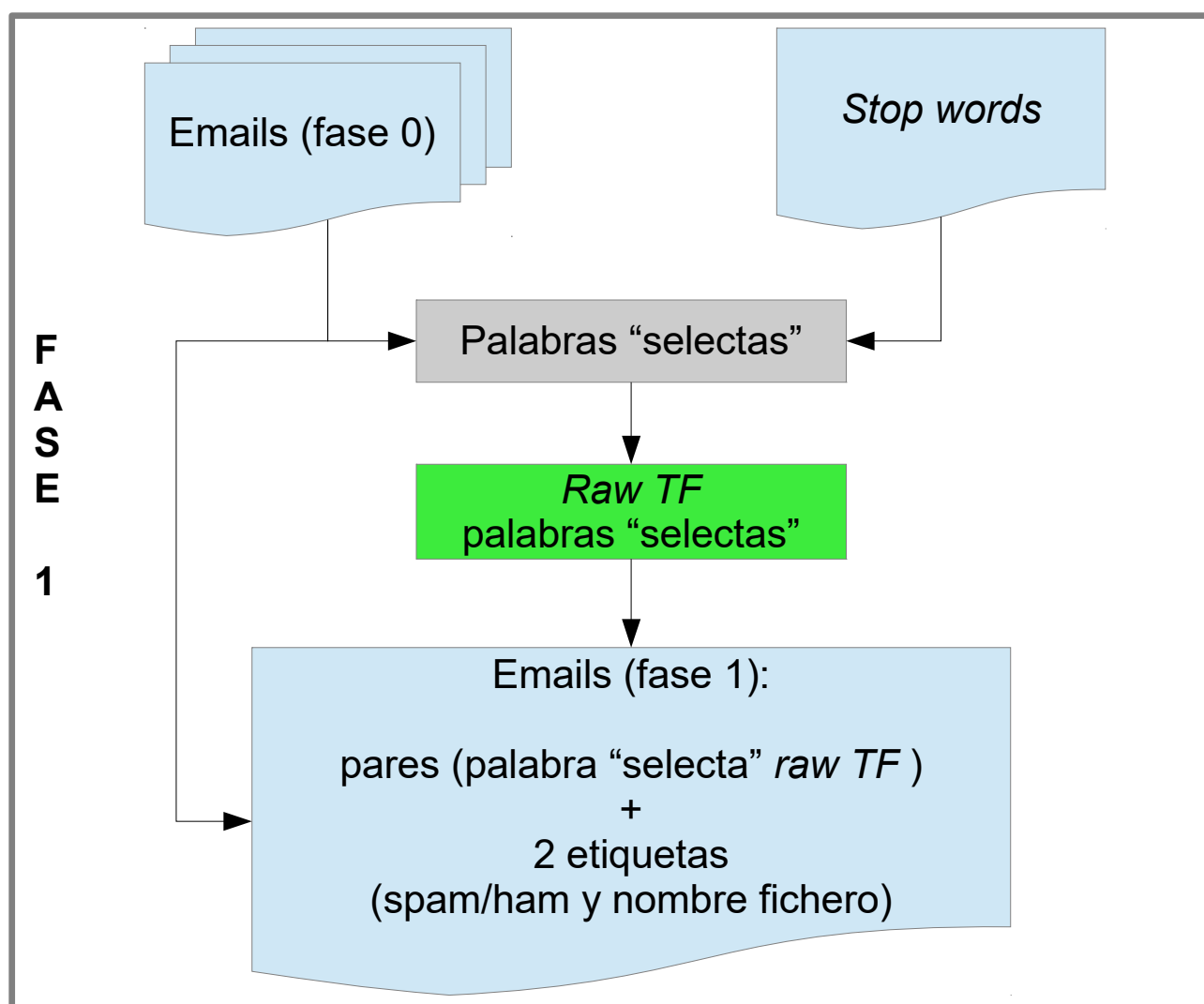


Figura 17: Diagrama de flujo de la fase 1 de nuestro preprocesado.

2.2.1.2.4.2 Fase 2 del preprocesado

Principalmente, en esta fase se calcula, para cada correo electrónico, cierto vector de acumuladores en el que cada componente se construye mediante la suma de las *raw TF* de todas las palabras de cada categoría presentes en el mensaje. Para ello se sigue la planificación expuesta a continuación y también esquematizada en la Figura 18.

Primeramente, se construyen y guardan todas las tablas de hash a través de las cuales se accederán a las palabras pertenecientes a cada una de las 13 categorías existentes, palabras que, como se había mencionado, estaban almacenadas en varios archivos de texto tras haber sido clasificadas manualmente.

Para cada mensaje obtenido del fichero de salida de la fase anterior, se acumulan las *raw TF* de todas sus palabras *selectas*, las cuales han sido buscadas en todas y cada una de las tablas de hash anteriores – puesto que una misma palabra puede pertenecer a varias categorías, o incluso a ninguna, en cuyo caso es ignorada de cara a posteriores análisis – con el fin de saber en qué categorías están. Con ello obtenemos 13 acumuladores que representan la cantidad de ocurrencias de palabras para cada una de las categorías de un correo concreto (Ej.: sean C1, C2 y C3 categorías de palabras, y sean P1 una palabra perteneciente a C1, y P2 otra perteneciente tanto a C1 como a

C2; si en un mensaje solamente aparecen 10 veces P1 y 5 veces P2, el acumulador relativo a C1 acumulará un valor de 15, el de C2 almacenará un 5 y el de C3 será un 0).

La salida de esta fase es un archivo de texto donde cada línea, que al igual que en la fase anterior también se referirá a un mensaje, está formada por un vector de números enteros, de tamaño 13 – que se corresponde con los contadores mencionados – seguido de dos etiquetas indicadoras de spam o ham – que la RNA obviará – y el nombre del fichero tratado, respectivamente, tal y como ya se habían utilizado con el mismo fin en la fase 1 precedente. Cada componente de dicho vector se guarda en una posición fija y en el orden ya indicado en un apartado precedente: Sexo / Relaciones, Medicina / Drogas / Cuerpo, Apuestas / Azar / Adivinación, Banca / Inversiones / Seguros / Compra-Venta, Enlace / Correo electrónico, Otros Idiomas, Ofuscación (mal escritos a propósito); Negocios / Empresa / Gobierno, Internet / Tecnología; Documentos / Trámites, Nombres / Familia, Turismo / Regiones y Fichero Adjunto.

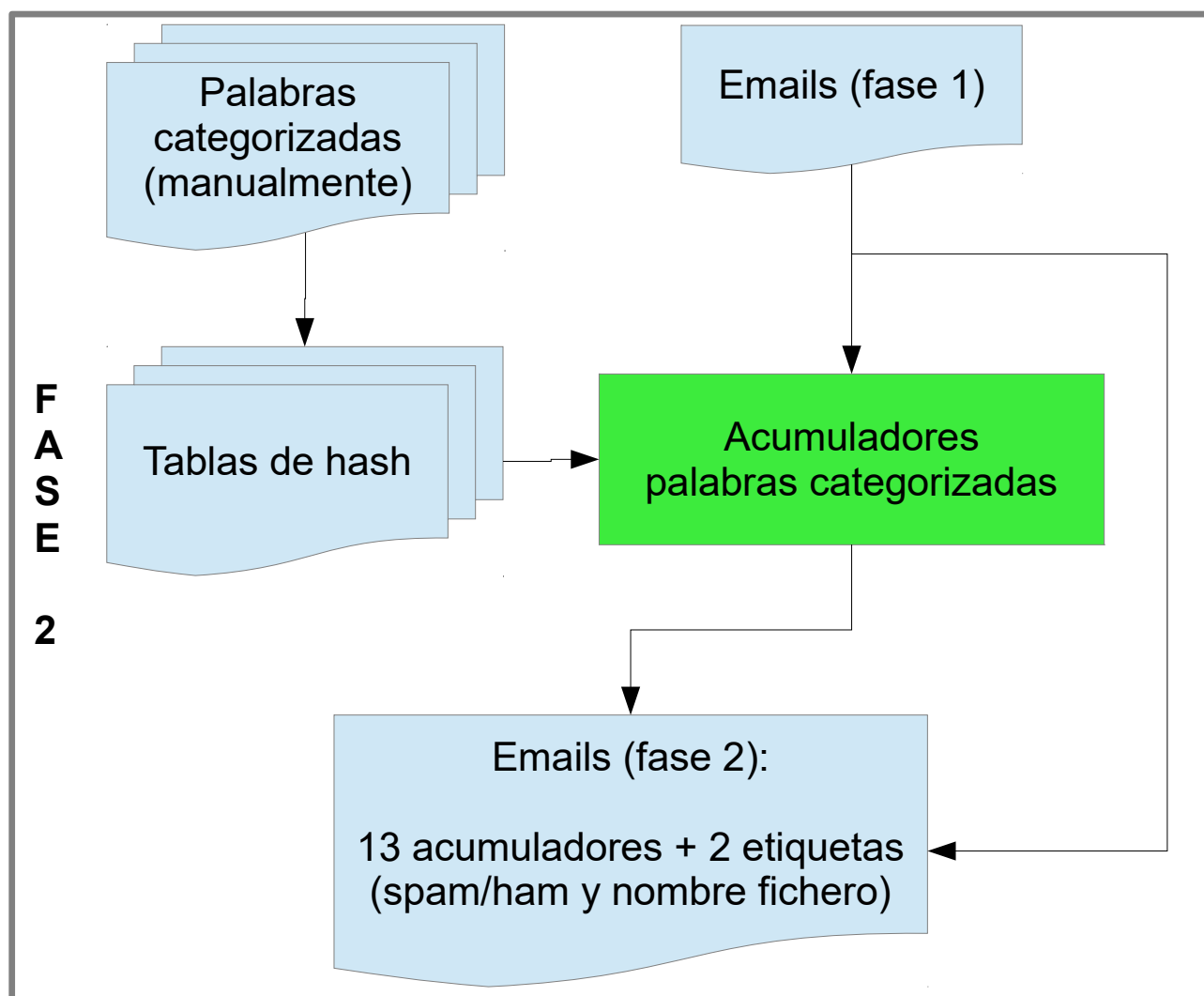


Figura 18: Diagrama de flujo de la fase 2 de nuestro preprocesado.

2.2.1.2.4.3 Fase 3 del preprocesado

Debido a la ya comentada, y probada, existencia de categorías o clases de spam, se ha encontrado que la opción de utilizar $TF*ICF$ en vez de $TF*IDF$ es más adecuada para ponderar las palabras dentro de cada categoría. De todas las versiones de $TF*ICF$ existentes, se ha decidido

combinar las dos siguientes, propuestas por estos autores:

- (Lertnattee y Theeramunkong 2004): cada una de las tres versiones de *ICF* (logarítmica, linear y raíz cuadrada) de dichos autores se han implementado con el fin de comprobar cuál obtiene mejores resultados. Por otra parte, el factor *TF* que ellos utilizan es sólo a nivel de categoría, mientras que el nuestro combinará éste con el *TF* a nivel de documento también utilizado por el siguiente autor.
- (Wang y Zhang 2013): modelo de ponderado de términos mixto ya que el factor *TF* del *TF*ICF* es estimado a nivel de documento mientras que el *ICF*, solamente logarítmica en este caso, lo es a nivel de categoría.

En resumidas cuentas, el *TF* a nivel de categoría (*TF_categoría*) se usará para ponderar las palabras dentro de cada categoría en conjunción con cada uno de los tres *ICF* con el fin de otorgar más importancia a palabras muy frecuentes dentro de esa categoría pero menos o nada frecuentes en las restantes. Por otro lado, el *TF* a nivel de documento (*TF_documento*), cuyo significado es idéntico al de la fase 2 del preprocesado, se utilizará como factor multiplicador de los tres *TF*ICF_categoría* (*TF_categoría* multiplicada por una de las tres variantes de *ICF_categoría*) anteriores de forma que tengan mayor relevancia las palabras que aparezcan más veces dentro de dicho documento. Por tanto y en resumidas cuentas, cada palabra dentro de cierto correo electrónico tendrá un valor asociado dado por las siguientes tres componentes:

- El *TF_documento* de dicha palabra en dicho correo electrónico, idéntico significado al *TF* de la fase 2 precedente.
- El *TF_categoría* de dicha palabra en cada una de las categorías donde esté almacenada.
- Una de las tres *ICF_categoría* para dicha palabra.

Para el cálculo de los tres tipos de *ICF* (concretamente para cada denominador) hace falta haber analizado todos los correos electrónicos disponibles (contar palabras *selectas* y calcular los acumuladores de palabras de cada categoría) o, dicho de otra forma, se puede hacer uso de los ficheros de salida de las fases 1 y 2 del preprocesado. En base a dichos acumuladores, se asocian las palabras de dicho correo electrónico a la categoría ganadora (por defecto sólo a 1), almacenándose tanto la palabra como su *TF_categoría*. Tras haber finalmente calculado los *ICF* tras lo anterior, se computan las tres *TF*ICF_categoría* para cada palabra dentro de cada correo electrónico, cuyo cálculo únicamente se diferencia en el *ICF* aplicado. Sobre dichos conjuntos de palabras ordenadas decrecientemente según su *TF*ICF_categoría*, obtenemos varias versiones según el porcentaje de palabras que deseamos coger, proceso de selección basado en un Top $k\%$ de porcentaje de tamaños, siendo $k \in \{10, 25, 50, 75, 90, 95, 100\}$, es decir, se eligen el Top $k\%$ de palabras con mayor *TF*ICF_categoría*.

Llegados a este punto, volvemos a analizar todos los correos electrónicos disponibles para construir su vector asociado. Dicho vector, similar al de la fase 2 pero en este caso es de números en coma flotante, también está conformado por 13 acumuladores, cuya construcción se consigue de la siguiente forma: para cada categoría, acumulan los *TF*ICF_documento* de todas sus palabras que aparezcan en dicho correo electrónico.

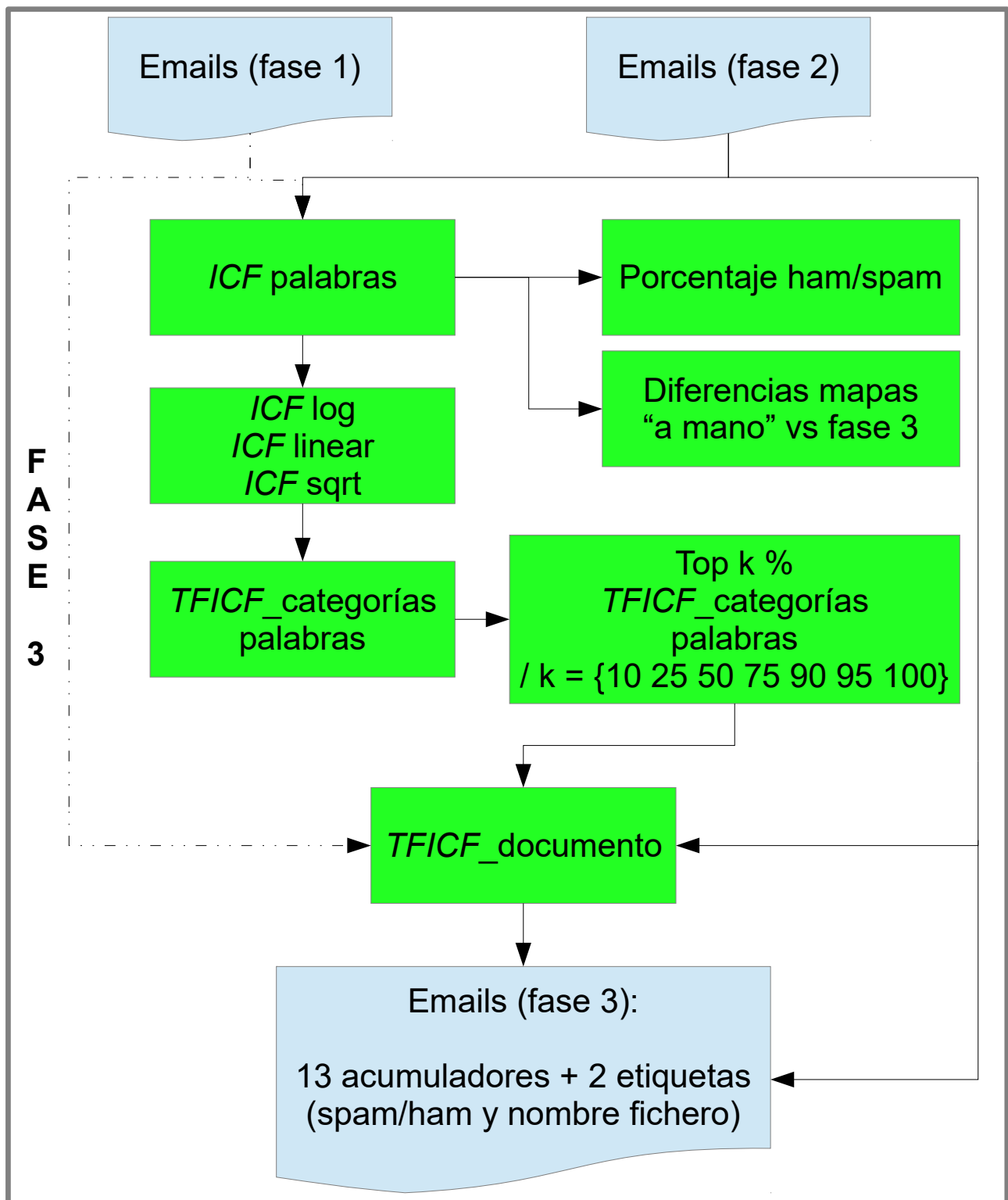


Figura 19: Diagrama de flujo de la fase 3 de nuestro preprocesado.

La salida principal de esta fase, véase la Figura 19, es un fichero de texto por cada tipo de *ICF* donde cada línea, también representando un correo electrónico, se corresponde con el anterior vector de 13 atributos y las mismas dos etiquetas ya mencionadas en las fases 1 y 2 anteriores. En uno de los dos escenarios que comprobaremos, explicados en mayor detalle a lo largo del Capítulo 3, el vector sufrirá una normalización previo a la entrada en la RNA para evitar que los atributos con menor amplitud de datos sean absorbidos por los que más tengan.

Durante la ejecución de esta fase también se ha calculado, y almacenado en otros ficheros, tanto los porcentajes de ham y spam para cada categoría como la comparación de pares de categorías entre el método manual y la fase 3 del preprocesado.

2.2.1.2.5 Categorización de palabras manual vs preprocesado de la fase 3: tamaños y diferencias

La categorización de palabras, o, dicho de otro modo, identificar a qué categorías se corresponde una determinada palabra, inicialmente se realizó manualmente y, posteriormente, se obtuvo durante la fase 3 del preprocesado, proceso ahora sí automatizado.

El procedimiento manual lo llevó a cabo el autor de este proyecto junto a su padre y consistió en, dada una lista con todas las palabras de longitud mayor a 2 presentes en todos los correos electrónicos, la identificación de las categorías a las que una palabra concreta pertenece más adecuadamente, de las 13 disponibles, teniendo en cuenta el uso y contexto. Es por ello que esta categorización conllevó, como ya se comentó, un alto coste temporal en el que, afortunadamente, no haría falta volver a incurrir por parte de los usuarios finales del sistema. Las ventajas de dicho procedimiento manual ya fueron expuestas en el apartado 2.2.1.2.2:

- Robustez frente a intentos de ofuscación de las palabras por parte de los *spammers*.
- No requerirse la aplicación de métodos de lematización sobre dichas palabras.

Por un lado, los resultados, relativos a las palabras contenidas en cada una de las categorías, obtenidos en la fase 3 del preprocesado (p3) con respecto a los obtenidos manualmente (pm) no dejan lugar a dudas de que, para todo par de categorías, existe mucha mayor cantidad de palabras en los de la fase 3 (p3) pese a que estos últimos sólo han hecho uso del 80% del cuerpo. Dichos resultados podemos observarlos en las Tabla 14 y Tabla 15.

Por otro lado, los resultados calculados como las diferencias entre ambos conjuntos de palabras según el procedimiento utilizado ($pm - p3$, y $p3 - pm$)⁸⁸ también confirman la observación anteriormente expuesta, véanse las Tabla 16 y Tabla 17.

88 Dados los conjuntos A y B, la diferencia de A menos B (o entre A y B) es otro conjunto $A - B$ (o también $A \setminus B$) cuyos elementos son todos aquellos elementos de A que no lo sean de B (en color amarillo en la Figura 20).

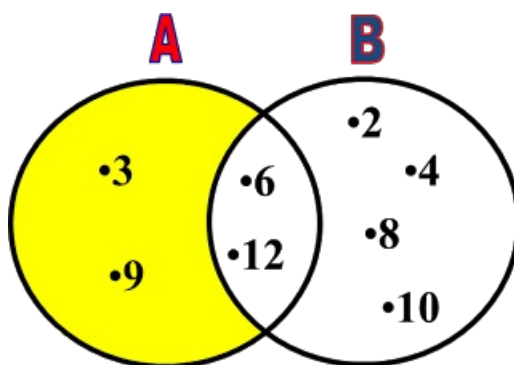


Figura 20: Diferencia entre los conjuntos A y B (en color amarillo).

Fuente: matematicaabelortega.blogspot.com.es

Tabla 14: Cantidades de palabras de las categorías 1-6 según procedimiento aplicado.

Categorías Procedimiento	Sexo	Medicina	Azar	Compras	Enlaces	Extranjeras
pm	263	1249	51	866	322	4300
p3	3142	13708	1079	27050	13773	20838

Tabla 15: Cantidades de palabras de las categorías 7-13 según procedimiento aplicado.

Categorías Procedimiento	Ofuscación	Negocios	Tecnología	Documentos	Nombres	Lugares	Adjuntos
pm	1253	247	136	75	247	156	18
p3	2547	13876	3988	3061	1945	937	1223

Tabla 16: Cantidades de palabras de las categorías 1-6 en la diferencia de los procedimientos.

Categorías Diferencia	Sexo	Medicina	Azar	Compras	Enlaces	Extranjeras
pm – p3	143	527	23	258	194	1267
p3 – pm	3022	12986	1051	26442	13645	17805

Tabla 17: Cantidades de palabras de las categorías 7-13 en la diferencia de los procedimientos.

Categorías Diferencia	Ofuscación	Negocios	Tecnología	Documentos	Nombres	Lugares	Adjuntos
pm – p3	1053	103	89	35	187	131	6
p3 – pm	2347	13732	3941	3021	1885	912	1211

Como ejemplo, en la Tabla 18 indicamos las palabras incluidas en las diferencias de conjuntos según el procedimiento utilizado de algunas de las categorías, las de menor tamaño, que hemos indicado en las Tabla 16 y Tabla 17.

Tabla 18: Palabras incluidas en las diferencias de conjuntos de algunas de las categorías, las de menor tamaño.

Categoría	Palabras
Adjuntos (pm – p3)	'attacehd' 'attachement' 'attahed' 'attatched' 'nsf' 'pps'
Azar (pm – p3)	'bankgirolotto' 'casiino' 'casinos' 'cerdaz' 'cerdazlottery' 'cerdazlotterysa' 'freelottolottery' 'freepicklotto' 'galilottery' 'horoscope' 'loterij' 'lott' 'lotterysa' 'lottos' 'lotz' 'powerball' 'powerballinfo' 'tattlotto' 'thecasinothemepark' 'unionlotteriesuk' 'unionlotteryuk' 'winningnotification' 'winnner'
Documentos (pm – p3)	'agreement' 'agreemeent' 'agreementat' 'agreementcdgt' 'agreemtent' 'archive' 'archived' 'archiving' 'certificates' 'certification' 'certifications' 'cheat' 'contraction' 'databased' 'docu' 'documentations' 'dossier' 'manuscript' 'manuscripts' 'memoranda' 'payrolls' 'paystub' 'propaedeutical' 'subcontracting' 'subcontracts' 'summaries' 'summarily' 'summarise' 'summarising' 'summarized' 'treatment' 'treatments' 'workorder' 'workorders' 'worksheets'

2.2.2 Etapa de procesado

Los sistemas detectores de correos no deseados propuestos en este trabajo se basan en arquitecturas neuronales no supervisadas y, más específicamente, en los mapas autoorganizados.

En general, y nuestro sistema no es una excepción, la estructura de las RNA se ve influida, entre otros, por las características propias del entorno de información donde opere, por el número de ejemplares que deba clasificar, y por las limitaciones impuestas por el entorno de trabajo, tanto hardware como software (Álvarez Romero, Domínguez Brito y Suárez Araujo 2013). Todos ellos actúan, en mayor o menor medida, como factores limitantes a la hora del diseño e implementación de sistemas basados en RNA. Es por ello que los hemos tenido muy en cuenta durante la realización de este proyecto y, mayormente, durante la etapa de procesado de los datos.

Como ya se detalló en la sección 2.2.1, tras el preprocesado de los datos, los correos electrónicos quedan definidos por 13 variables correspondientes a las 13 categorías existentes. Cuanta mayor es la cantidad de variables de entradas disponibles, mayor necesidad de cómputo requerirá el proceso de aprendizaje de las RNA. Ello implica que el preprocesado, aparte de proporcionar una mejor representación de la información (lo que ayuda a realzar el modelo de representación interna de las redes), también aporta mayor velocidad de cómputo en los extensos procesos de aprendizaje. Este coste computacional podemos encontrarlo en el apartado 3.3.2.1.

Las características que presentan las diferentes configuraciones de *SOM* analizadas vienen detalladas y justificadas en profundidad al comienzo del Capítulo 3, seguidas de los resultados obtenidos con las diversas métricas implementadas.

2.3 Entorno de trabajo

En esta sección describiremos las distintas herramientas software empleadas durante la elaboración de este proyecto, finalizando con una mención al soporte hardware sobre el que se ha desarrollado.

2.3.1 MATLAB

Es un entorno de cálculo técnico, multiplataforma, de altas prestaciones para cómputo numérico y visualización, que integra, entre otros, análisis numérico, cálculo matricial, procesamiento de señales y gráficos. Ha sido desarrollado por la corporación privada MathWorks desde 1984. En nuestro proyecto hemos hecho uso de la versión 7.12.0 (R2011a) sobre Windows 7 *Home Premium* (64 bits) *Service Pack 1* y, posteriormente, sobre Windows 10 *Home*.

MATLAB puede ser dividido en cinco partes:

- Lenguaje MATLAB: de alto nivel, interpretado y especializado en vectores y matrices (de ahí que el nombre sea el *portmanteau* *MATrix LABoratory* o laboratorio matricial). Cuenta con características comunes a otros lenguajes parecidos tales como instrucciones de control de flujo, funciones, estructuras de datos, programación orientada a objetos, entrada/salida...

- Entorno de trabajo MATLAB: conforma un *IDE*⁸⁹, el cual cuenta con editor de código fuente, esquema de colores, depurador, explorador de ficheros, etc.
- Manejador de gráficos: una de las grandes ventajas de MATLAB es la sencillez y diversidad de operaciones que se pueden realizar con gráficos, tanto en su creación como en su personalización. También permite desarrollar interfaces gráficas de usuario.
- Librería de funciones matemáticas: amplia colección de algoritmos computacionales de distinto nivel de complejidad.
- *API*⁹⁰ de MATLAB: librería de subprogramas o métodos que permiten el desarrollo de aplicaciones abstrayendo el modo de implementarlas al únicamente expresarlas como operaciones, entradas, salidas y tipos de datos. Entre otras funcionalidades, esta *API* permite utilizar programas escritos en algunos lenguajes de programación (*i.e.* C, C++, Java, Fortran, .NET, Perl y Python) que interactúan con MATLAB de distintas formas tales como la ejecución dentro de MATLAB, el manejo de ficheros de datos de MATLAB (con extensión MAT), etc.

Por otro lado, aunque MATLAB dispone de miles de comandos y funciones, no incluye todas las deseables. Es por ello que tanto MathWorks – creadora de MATLAB – como otras empresas y usuarios han desarrollado algunas colecciones de funciones que extienden el entorno de MATLAB para resolver ciertas clases particulares de problemas (Ej.: RNA, procesamiento de señales, simulación, lógica difusa, robótica, etc). Estas colecciones, no necesariamente libres ni gratuitas, se denominan *toolboxes* y, en los siguientes dos subapartados, expondremos las que hemos utilizado en nuestro proyecto.

Una desventaja que presenta este entorno de desarrollo es que se trata de software privativo y, según el tipo de licencia, no muy económico⁹¹. Pese a ello, es de los más usados en ámbitos tan variados como la ingeniería, la ciencia y la economía sobre todo por su estabilidad, capacidades, documentación, y cantidad y variedad de *toolboxes*.

2.3.2 SOM Toolbox

Se trata de una *toolbox*, libre, para MATLAB (Vesanto 1999; Vesanto et al. 1999, 2000 ; Aalto University (CIS) 2005) basada en SOM_PAK⁹² de forma que *SOM Toolbox* combina las capacidades gráficas del entorno MATLAB con las bondades de SOM_PAK, superando su cantidad de funciones y llegando a ser incluso más rápida aunque no tan eficiente en el consumo de memoria. En ciertos casos (Vesanto et al. 1999) ha sido requerida de 2 a 10 veces más cantidad, según se use el algoritmo *batch* o el secuencial en la *toolbox* respecto al de la SOM_PAK – aún cuando el algoritmo de la propia *toolbox* sea sustituido por el de la SOM_PAK y lanzado internamente. En nuestro proyecto hemos hecho uso de la versión 2.0 de 17 de Marzo de 2005.

Tanto SOM_PAK, desde 1992, como *SOM Toolbox*, desde 1997, fueron desarrolladas en el Laboratorio de Ciencias de la Información y de la Computación de la *Helsinki University of Technology* (Finlandia), participando el propio Teuvo Kohonen en el desarrollo de la primera de estas aplicaciones. Por otro lado, *SOM Toolbox* es también recomendada por (Kohonen 2001,

89 *Integrated Development Environment* o Entorno de Desarrollo Integrado.

90 *Application Program Interface* o Interfaz de Programación de Aplicaciones.

91 El rango de precios de MATLAB varía bastante: desde 35€ (versión más básica, sólo para estudiantes) hasta 2000€ (licencia comercial individual estándar).

92 SOM_PAK es una aplicación gratuita, implementada en C (Kohonen et al. 1996), para UNIX y DOS originalmente. Es bastante completa y rápida pero tosca en el apartado de visualización de los datos y resultados.

2013 ; Kohonen, Nieminen y Honkela 2009), entre otros investigadores (Westerlund 2005; Uriarte y Martín 2005; Sassi 2011).

2.3.3 Parallel Computing Toolbox

Esta *toolbox* para MATLAB (MathWorks 2014) permite resolver problemas computacionalmente intensivos usando procesadores multinúcleos, multiprocesadores, *GPUs*⁹³ y clústers de ordenadores. Mediante ciertas construcciones de alto nivel – Ej.: bucles *for* paralelos (denominados "*parfor*"), tipos de matrices especiales, y algoritmos numéricos paralelizados – es posible paralelizar aplicaciones de MATLAB sin necesidad de programación *CUDA*⁹⁴ o *MPI*⁹⁵. También permite la ejecución de múltiples simulaciones de Simulink en paralelo. Las aplicaciones se pueden ejecutar de forma interactiva o en modo *batch*. No debemos confundir la ejecución en paralelo que ofrece esta *toolbox* con la ejecución multihilos, inherente y automáticamente aplicada en muchas funciones de MATLAB y de otras *toolboxes*. Hemos utilizado la versión 5.1.

Debido a ciertas limitaciones de esta *toolbox*⁹⁶, no siempre fue posible utilizarla aunque, cuando sí lo fue, nos permitió aprovechar la máxima capacidad de procesamiento de nuestro multiprocesador multinúcleo. Ello se ha conseguido con la ejecución de las aplicaciones sobre *workers*⁹⁷ que corren localmente, máximo uno en cada núcleo. Sin cambios en el código se podría haber ejecutado incluso sobre un clúster de ordenadores o sobre un servicio de computación *grid*⁹⁸ – a través del servidor de computación distribuida de MATLAB (*MATLAB Distributed Computing Server*), el cual también es de pago.

En nuestro caso, la ventaja de haberla empleado es que conllevó una importante reducción del coste computacional temporal⁹⁹ durante la etapa de aprendizaje de las diferentes configuraciones de *SOM* ya que, a grandes rasgos, podríamos interpretarlo como si hubiéramos contado con mayor número de ordenadores con los que trabajar. (Oberg et al. 2008) aducen que la mejora del

93 *Graphics Processing Unit* o Unidad de Procesamiento Gráfico. Es un coprocesador dedicado al procesamiento de gráficos u operaciones de coma flotante, para aligerar la carga de trabajo del procesador central (CPU) en aplicaciones como los videojuegos o aplicaciones 3D interactivas.

94 Son las siglas de *Compute Unified Device Architecture* (Arquitectura Unificada de Dispositivos de Cómputo) que hace referencia tanto a un compilador como a un conjunto de herramientas de desarrollo creadas por nVidia que permiten a los programadores usar una variación del lenguaje de programación C – u otro mediante adaptadores – para codificar algoritmos en *GPUs* de nVidia. Básicamente, es una arquitectura para la computación en paralelo sobre *GPUs* de dicho fabricante.

95 *Message Passing Interface* o Interfaz de Paso de Mensajes. Es un estándar que define la sintaxis y la semántica de las funciones contenidas en una biblioteca de paso de mensajes diseñada para ser usada en programas que exploten la existencia de múltiples procesadores. Su principal característica es que no precisa de memoria compartida, por lo que es muy importante en la programación de sistemas distribuidos.

96 La limitación más importante y restrictiva con la que nos encontramos fue la de que, cuando se use la instrucción "*parfor*", las iteraciones deben ser completamente independientes entre sí. Es por ello que solamente pudimos utilizarla durante el aprendizaje de cada configuración de *SOM* puesto que no son operaciones interdependientes.

97 Motores computacionales de MATLAB los cuales podríamos considerarlos como entornos de ejecución de MATLAB corrientes pero carentes de interfaz gráfica de usuario.

98 Infraestructura que permite la integración y el uso colectivo de ordenadores de alto rendimiento, redes y bases de datos que son propiedad y están administrados por diferentes instituciones (control no centralizado). Los recursos pueden ser heterogéneos y estar distantes geográficamente (lo común es que sea en diferentes continentes) por lo que se comunican mediante redes de área extensa como Internet.

99 Debido a que se usaron únicamente 4 de los 8 núcleos de la CPU, estimamos una mejora del tiempo de cómputo de un 75%, es decir, lo que antes se tardaría 4h, ahora sería de 1h aproximadamente. (Oberg et al. 2008) demuestran que dicha mejora suele ser inferior a la lineal puesto que el uso de "*parfor*" incurre en *overhead* o penalizaciones de tiempo debidas a la necesidad de transmitir las variables, las instrucciones y los datos entre los diferentes *workers*.

rendimiento proporcionada por "*parfor*" es mayor cuando se incrementa tanto el número de *workers* como la cantidad de iteraciones realizadas.

2.3.4 Hardware

Este proyecto se ha ejecutado sobre un ordenador portátil DELL XPS L702X cuyas características más relevantes, más propias de un ordenador de sobremesa que de un portátil, son:

- Procesador Intel Core i7-2760QM CPU @ 2,40 Ghz, con 4 núcleos físicos (cada uno de los cuales con 2 núcleos lógicos) y 6 MB de memoria caché L3 compartida.
- Memoria RAM DDR3 con un tamaño total de 12 GB.
- 2 discos duros internos con una capacidad total de 1 TB, además de 1 TB en un disco duro externo, utilizado para backups (tarea compartida con Dropbox, un servicio de almacenamiento de archivos en la nube). En ambos discos duros el nivel de ocupación previo al desarrollo del proyecto rondaba el 75% aproximadamente.

Capítulo 3. Desarrollo de filtro antispam basado en mapas autoorganizados de Kohonen. Implementación y resultados

Introducción: En este capítulo se explica el desarrollo e implementación del filtro antispam basado en los mapas autoorganizados de Kohonen, concretamente la implementación de la fase de procesamiento del sistema inteligente propuesto. Se presentarán las diferentes configuraciones de *SOM* utilizadas durante el desarrollo de dicho sistema, los resultados obtenidos y la discusión de los mismos, hasta alcanzar el filtro antispam definitivo.

3.1 Características de la red utilizada

3.1.1 Características de la arquitectura neuronal utilizada

Como ya se ha mencionado en el apartado 1.4, la arquitectura neuronal utilizada en este proyecto ha sido el SOM, una Red Neuronal Artificial no supervisada¹⁰⁰, con aprendizaje competitivo.

Las diversas configuraciones de SOM analizadas durante este capítulo comparten algunas características, muchas de las cuales se basan en recomendaciones de distintos autores. Estas recomendaciones se exponen, parafraseadas en algunos casos, a continuación, agrupadas según la autoría:

- (Kohonen 2001, 2013):
 - “Las rejillas hexagonales son mejores, más rápidas y más ilustrativas [que las rectangulares]”: todas las configuraciones de SOM probadas utilizan rejillas (opción “*lattice*” en la *SOM Toolbox*) hexagonales.
 - “Alrededor de 50 ítems [correos electrónicos en nuestro caso] por nodo son recomendables”: debido a ello se ha probado el mapa de Kohonen con tamaño 13×13 ¹⁰¹.
 - “El algoritmo [de entrenamiento] en modo *batch* es mejor, más rápido¹⁰² y puede ser generalizado para datos no vectoriales, aunque trae aparejado un consumo de memoria bastante alto”: fue el finalmente seleccionado (función “*som_batchtrain*” de la *SOM Toolbox*) en contraposición al algoritmo secuencial (“*som_seqtrain*”), ambos explicados en el Capítulo 1.
 - “La inicialización de los pesos en el SOM es importante”: en nuestro caso, hemos aplicado inicialización lineal en base a los mayores autovectores¹⁰³ – mediante la función “*som_lininit*” de la *SOM Toolbox* – en vez de la aleatoria – “*som_randinit*” – por rapidez de ejecución y convergencia.
 - “El tamaño de la vecindad debe ser elegido bastante amplio al comienzo (incluso superior a la mitad del tamaño del SOM) – para que el mapa se ordene globalmente y no localmente en forma de mosaico¹⁰⁴ – decrementándose paulatinamente hasta la unidad durante las primeras 1000 iteraciones. Durante la fase de convergencia puede mantenerse el tamaño 1”.
 - “El ratio de aprendizaje (α o $\alpha(t)$) debe empezar con un valor cercano a la unidad y decrecer monótonamente – pero siempre sobre 0,1 – durante los 1000 primeros pasos, siendo la forma de esa variación no importante (Ej.: $\alpha(t) = 0,9 (1 - t / 1000)$). El ordenado topológico ocurre en las fases iniciales (fases de ordenación) mientras que en las siguientes iteraciones, numerosas, se produce el ajuste fino del mapa (fase de convergencia). En esta fase, y durante bastantes (miles) iteraciones, es recomendable que el valor de $\alpha(t)$ sea bajo (menor o igual a 0,01)”. El algoritmo de entrenamiento *batch*, finalmente utilizado por las ventajas

100 La propia *SOM Toolbox* dispone de una versión del SOM basada en aprendizaje supervisado. También existe este tipo de SOM en los entornos de desarrollo R y WEKA.

101 La cantidad de 13 se obtiene como el redondeo al alza de la operación $\sqrt{(7933 \div 50)}$ siendo 7933 la cantidad de correos electrónicos preprocesados en el subconjunto entrenamiento-ajuste.

102 (Wilson y Martinez 2003) afirman todo lo contrario siempre que se den ciertas condiciones *e.g.* grandes conjuntos de entrenamiento.

103 También conocidos como vectores propios, vectores característicos o eigenvectores (v). Entre otros usos, pueden indicar las características más relevantes de cualquier conjunto de datos, o las direcciones a las que tienden una nube de puntos. Este concepto está bastante relacionado con los de autovalor, valor propio, valor característico o eigenvalor (λ ; escalar), según la fórmula (siendo A una matriz cuadrada) $A v = \lambda v \rightarrow (A - \lambda I) v = 0$

104 Pequeñas formaciones ordenadas localmente pero que globalmente no lo están puesto que la dirección del orden cambia discontinuamente.

anteriormente expuestas, no contiene parámetro relativo al ratio de aprendizaje¹⁰⁵, de ahí que esta recomendación no sea aplicable. Sin embargo, este parámetro sí aparece con el algoritmo de entrenamiento secuencial.

- “Que una palabra tenga bajo *IDF* suele implicar que sea una *stop word* y, por tanto, pueda ser ignorada”: esta recomendación fue tenida en cuenta durante el preprocesado de los datos (Capítulo 2).
- (Leow 2014):
 - Función de vecindad gaussiana: el valor máximo se encuentra en la neurona ganadora, y decrece a medida que se aleja de ésta (Figura 8).
 - Mapas 2-D cuadrados con forma de hoja (*sheet*).
- (Gabrielsson y Gabrielsson 2006):
 - “El número de pasos [o iteraciones] durante el entrenamiento debe ser razonablemente grande, al menos 500 veces el número de nodos de la red, para una buena exactitud estadística”. Como se había mencionado en el apartado 1.4, debido al uso del algoritmo de entrenamiento *batch* no se han utilizado pasos sino *epochs*, por lo cual se presenta la totalidad del conjunto de entrenamiento a la RNA antes de realizar la modificación de los pesos de la red, para su aprendizaje. Aún así, se han hecho los experimentos con cantidades altas de estos (hasta 8000 *epochs* en nuestro proyecto) de forma que se supera en varios órdenes de magnitud la cantidad de pasos equivalente recomendada.

3.1.2 Conjuntos de datos de entrada

El conjunto de datos de entrada para el *SOM*, obtenidos preprocesando el cuerpo de datos original de la forma ya indicada en el Capítulo 2, fueron particionados en tres subconjuntos disjuntos¹⁰⁶ de datos de entrenamiento, ajuste paramétrico y testeo (Keller 2003; Borovicka et al. 2012; Karahoca 2012) (Figura 14), con el objetivo de obtener la configuración óptima de la arquitectura *SOM* de la fase de procesamiento del filtro antispam propuesto. El 80% del cuerpo de datos corresponderá con el conjunto de entrenamiento y de ajuste paramétrico, en la proporción 60% y 20%, respectivamente. El 20% restante se dedica al conjunto de testeo. Estos subconjuntos de datos tienen la siguiente función:

- Subconjunto de entrenamiento. Se utilizará para entrenar cada una de las variantes o configuraciones de RNA que se decidan comprobar.
- Subconjunto de ajuste paramétrico. Permitirá afinar la arquitectura neuronal. Se podrá analizar la calidad de las diversas configuraciones diseñadas seleccionando las que podríamos denominar como conjunto de posibles soluciones finales (Tabla 19).
- Subconjunto de testeo. Confirmará la calidad del susodicho conjunto de posibles soluciones finales, probando la configuración óptima sobre los datos de este subconjunto, los cuales no han sido vistos anteriormente.

El subconjunto “entrenamiento-ajuste”, “*Train-Valid*” o “*TV*” debido a que se corresponde con la fase de entrenamiento y ajuste-validación del *SOM*, es el que se usó para obtener las mejores configuraciones, cuyos resultados son expuestos a lo largo del apartado 3.3. En nuestro caso, dicho 80% del corpus suma la cantidad de 7933 correos electrónicos, repartiéndose en cada uno de esos dos subconjuntos de la siguiente forma: 5950 para entrenamiento y 1983 para ajuste. Algo similar ocurre para la fase de testeo (en adelante “*Test*” o “*T*”) cuyo subconjunto es del mismo tamaño y

105 Ya que, por un lado, se indica explícitamente en (Gabrielsson y Gabrielsson 2006) – concretamente al final del apartado 4.11 de dicho documento – y, por otro lado, no aparece como argumento de entrada, ni en la documentación ni en el código, de la función “*som_batchtrain*” de la *SOM Toolbox* utilizada.

106 En matemáticas, dos conjuntos son disjuntos si su intersección es vacía, es decir, no tienen elementos en común.

porcentaje que el de ajuste. Para todos estos subconjuntos la cantidad de ham y spam está balanceada según la ratio 1:1¹⁰⁷.

Como se verá en los apartados siguientes, se han estudiado dos escenarios distintos según se haya aplicado o no algún tipo de normalización sobre los vectores de entrada de la RNA: datos sin normalizar (escenario 1) y datos normalizados (escenario 2). Dicho tipo de normalización viene definido en el apartado 3.3.1.

3.2 Descripción de funciones de los *toolboxes* utilizadas

3.2.1 Del *SOM Toolbox*

Los siguientes subprogramas de la citada *toolbox* fueron utilizados en el orden mostrado, siendo los dos últimos los únicos no utilizados durante el entrenamiento de la RNA:

- “*som_read_data*”: almacena los datos preprocesados, leídos desde un fichero convenientemente formateado (descrito en el Capítulo 2), en una estructura de datos manejable por dicha *toolbox*.
- “*som_modify_dataset*”: permite añadir a una estructura de datos, o quitar de ésta, tanto atributos/características como ejemplares. Lo hemos utilizado para extraer ejemplares y obtener las estructuras de datos para entrenamiento, ajuste y testeo con el formato adecuado.
- “*som_normalize*”: normaliza, o renormaliza, una estructura de datos según el método de normalización que se le indique (parámetro opcional).
- “*som_init*”: inicializa un *SOM* linealmente en base a sus mayores autovectores. Le hemos pasado por parámetro tanto la forma y tamaño del mapa como la forma de la rejilla.
- “*som_batchtrain*”: utiliza el algoritmo *batch* para el entrenamiento del mapa de Kohonen. La función de vecindad, el número de *epochs* durante el cual entrenarse y el nivel de *tracking* han debido ser definidos.
- “*som_autolabel*”: autoetiquetado, o eliminado de etiquetas, de los datos sobre un mapa mediante el algoritmo de etiquetado que se le pase.
- “*som_bmus*”: encuentra las *BMU* (en cierta cantidad o con ciertas características) en un mapa para unos determinados vectores o estructuras de datos de entrada.
- “*som_quality*”: calcula el *MQE* y el *TE* de un mapa para ciertos datos o estructura de datos.

3.2.2 Del *Parallel Computing Toolbox*

Para acelerar la ejecución de las pruebas, en nuestro caso únicamente aplicable durante la fase de aprendizaje¹⁰⁸, se utilizó programación en paralelo mediante las siguientes funciones de dicha *toolbox*:

- “*matlabpool*”: permite las características de lenguaje paralelo en el lenguaje MATLAB a través de la activación de una tarea paralela que conecte un cliente de MATLAB con cierto número de *workers* o *labs* (instancias de MATLAB ejecutadas como esclavos a petición del cliente MATLAB maestro), pudiendo estar estos en otros ordenadores remotos. Sirve tanto para indicar o saber el número de *labs* como para abrir y cerrar dicha tarea paralela.
- “*parfor*”: se trata de un bucle *for* capaz de, siempre que existan otros *labs* correctamente

¹⁰⁷ Ratio 1:1 rozando la perfección, existiendo 3 mensajes de spam más que de ham.

¹⁰⁸ Debido a las limitaciones de la instrucción “*parfor*” que se explican más adelante en este mismo subapartado.

configurados y activos, ejecutarse en paralelo. Lamentablemente presenta limitaciones en tanto en cuanto las instrucciones dentro del bucle deben ser completamente independientes entre sí, lo cual puede ser difícilmente sino imposible de conseguir.

3.3 Implementación del filtro antispam. Resultados y discusión

3.3.1 Parámetros del SOM y del preprocesado

Por un lado, hemos estudiado los siguientes parámetros del SOM en las diversas configuraciones (1260 en total, 630 para cada uno de los dos escenarios), basándonos en decisiones mayormente fundamentadas en las recomendaciones de otros autores (apartado 3.1.1):

- Inicialización de los pesos de la RNA: en todas las configuraciones se ha elegido la opción del *SOM Toolbox* correspondiente a la inicialización lineal. La otra opción no usada, inicialización aleatoria (función “*som_randinit*” de la *SOM Toolbox*), hubiera traído aparejada la prueba de la misma configuración de SOM – con todo el resto de parámetros idénticos – varias veces para hacer frente a dicha aleatoriedad de los pesos iniciales.
- Normalización de los datos de entrada: decidimos probar si el hecho de aplicar normalización facilitaba la labor de clasificación y se obtenían mejores resultados. El tipo de normalización elegido ha sido el recomendado por defecto en la *SOM Toolbox* sobre MATLAB: opción “*var*”, es decir, varianza normalizada a la unidad.
- Número de *epochs*: se han modificado los valores de *epoch* (Sinha et al. 2010) dentro del siguiente rango: 100, 500, 1000, 3000, 5000 y 8000. Es obligado decir que, debido al altísimo coste temporal que supondrían cantidades mayores – en base al observado con 5000 *epochs* sobre un tamaño de la red de 50x50: cercano a las 12 horas de ejecución ininterrumpida – decidimos limitarnos a dicho conjunto de valores.
- Forma de la rejilla: las rejillas hexagonales fueron las seleccionadas para todas las pruebas en base a la recomendación de (Kohonen 2001, 2013).
- Forma del mapa: únicamente hemos probado con la más común que es en forma de hoja (opción “*sheet*”).
- Tamaño del mapa: viene indicado por la cantidad de neuronas en un lado del SOM (opción “*msize*” de la *SOM Toolbox*), variando entre los valores siguientes: 13, 20, 25, 40 y 50, es decir, SOM cuadrados desde 13x13 hasta 50x50. A mayor número de neuronas en la red, mayor coste computacional, de ahí que nos hayamos limitado a este conjunto.
- Autoetiquetado de los datos: debido a que los datos están originalmente etiquetados¹⁰⁹, podemos hacer uso de métodos de autoetiquetado de los datos para luego poder calcular las métricas dependientes de la existencia de etiquetas (indicadas en el apartado 1.4.3). Dichos métodos, obviamente, sólo son aplicables tras el aprendizaje de la red. Concretamente, hemos elegido el método de autoetiquetado de tipo frecuencia (opción “*freq*” de la función “*som_autolabel*” en la *SOM Toolbox*) ya que retorna el número de veces que aparece cada etiqueta – en nuestro caso, sólo de la primera de las dos que hay para cada mensaje: ham y spam; Ej.: ham (32) spam (1) – en cierta neurona de la red. El orden de las etiquetas ham y spam, en caso de que ambas existan, varía entre neuronas puesto que la anterior función de la *toolbox* usando “*freq*” las retorna ordenadas de manera descendente según su frecuencia de manera que la primera siempre se corresponderá con la más numerosa. No se ha hecho uso del método por votación (opción “*vote*”), más común, en la que sale ganando la etiqueta más votada porque se ignoran las restantes etiquetas y no sería de fácil aplicación el

¹⁰⁹ Como se indicó a lo largo del Capítulo 2, cada mensaje de correo electrónico se ha preprocesado para que venga acompañado de dos etiquetas: spam/ham, y nombre del mensaje de correo electrónico.

etiquetado no automático de los datos que a continuación se explica.

- Etiquetado no automático de los datos: tras el autoetiquetado anterior, con el que habremos obtenido la frecuencia de cada etiqueta en una neurona dada, debemos aplicar un etiquetado no automático de los datos con el fin de obtener varios puntos con los que graficar las curvas *ROC*¹¹⁰. Ello lo conseguimos modificando los valores de la matriz de confusión (*TP*, *FP*, *TN* y *FN*) y, por ende, los valores de especificidad y sensibilidad. Para lo cual, de entre muchos otros métodos existentes (Slaby 2007a, 2007b), hemos decidido hacer uso de *thresholds* o umbrales conjuntamente con el ratio de spam. De esa manera, si el umbral tratado en ese momento es inferior a dicho ratio, indica que existe cierto grado de incertidumbre o falta de confianza en la elección de la etiqueta de spam por lo cual se hará uso de la ham. Por el contrario, si el ratio de spam es superior o igual al umbral, se usaría la etiqueta de spam. Para obtener curvas *ROC* con cierto nivel de detalle, hemos decidido hacer uso de 100 *thresholds* (102, de 1 a 101, si tenemos en cuenta los valores extremos).

Por otro lado, hemos analizado los siguientes parámetros relativos al preprocesado, anteriormente explicados a lo largo del Capítulo 2:

- *ICF*: hemos hecho pruebas con las tres versiones implementadas: *log*, *linear* y *sqrt* (Lertnattee y Theeramunkong 2004).
- Top *k* porcentaje de palabras: hicimos pruebas con los siguientes Top *k*%, siendo $k = \{100, 95, 90, 75, 50, 25, 10\}$. El Top 100% significa que se usa la totalidad de las palabras *selectas* encontradas, es decir y generalizando, que elegimos el *k*% de las palabras con mayor puntuación. Dichas palabras fueron ponderadas y ordenadas descendientemente durante la fase 3 del preprocesado (apartado 2.3.2.4.3).

3.3.2 Resultados y discusión

Comenzaremos mostrando, tanto de forma tabulada como gráfica, los resultados obtenidos, seguida de una discusión e interpretación de los mismos. Finalizaremos con un subapartado donde se muestran los tiempos de ejecución y las funcionalidades de algunas de las funciones propias implementadas.

Antes de exponer los resultados, conviene explicar la nomenclatura utilizada durante todo este documento para la denominación de las diferentes configuraciones de *SOM* probadas. Para ello utilizaremos notación BNF¹¹¹, seguido de un ejemplo y una definición de cada componente de dicho nombre.

```
<nombre_config_SOM> ::= <nombre_BMU> <nombre_Top> <nombre_ICF> <tipo_vecindad>  
<forma_rejilla> <forma_mapa> <tamaño_mapa> <num_epochs> <nombre_carpeta> <num_intento>  
  
<nombre_BMU> ::= "BmusTrainValid" | "BmusTrainValidNorm" | "BmusTrainTest" | "BmusTrainTestNorm"  
  
<nombre_Top> ::= <espacio> "Top" <espacio> <k_Top> <espacio> "porc"
```

110 Originalmente, cada una de las diferentes configuraciones de *SOM* retorna un único valor de especificidad y sensibilidad, es decir, se trataría de clasificadores discretos por lo que solamente sería posible graficar un punto en el espacio *ROC* (o curva *ROC* si éste se utiliza conjuntamente con los puntos (0, 0) y (1, 1) (Fawcett 2004, 2006 ; Slaby 2007b; Wikipedia 2013)).

111 La notación de Backus-Naur, del inglés Backus-Naur *Form* (BNF), Backus-Naur *Formalism* o Backus *Normal Form*, es un metalenguaje que sirve para la definición de gramáticas libres de contexto y es usualmente empleada para describir la sintaxis de lenguajes de programación, protocolos de comunicación, formatos de documentos, y conjuntos de instrucciones. Ciertas variantes y extensiones de BNF, tales como la *Augmented Backus-Naur Form* (ABNF) y la *Extended Backus-Naur Form* (EBNF), cuentan con su propia documentación.

<k_Top> ::= "100" | "95" | "90" | "75" | "50" | "25" | "10"
 <nombre_ICF> ::= <esp_guion_esp> "ICF" <espacio> <opcion_ICF> <esp_guion_esp>
 <opcion_ICF> ::= "log" | "linear" | "sqrt"
 <tipo_vecindad> ::= "gaussian" <espacio>
 <forma_rejilla> ::= "hexa" <espacio>
 <forma_mapa> ::= "sheet" <espacio>
 <tamaño_mapa> ::= "13x13" | "20x20" | "25x25" | "40x40" | "50x50"
 <num_epochs> ::= <espacio> <aux_num_epochs> <espacio> "epochs"
 <aux_num_epochs> ::= "100" | "500" | "1000" | "3000" | "5000" | "8000"
 <nombre_carpeta> ::= <espacio> "mini mix"
 <num_intento> ::= <esp_guion_esp> "1"
 <esp_guion_esp> ::= <espacio> "-" <espacio>
 <espacio> ::= " "

Ej.: BmusTrainValidNorm Top 100 porc - ICF linear - gaussian hexa sheet 50x50 8000 *epochs* mini mix - 1

1. Nombre de las *BMU*: indica cómo se construyeron dichas *BMU*, es decir, qué mapa y qué estructura de datos se utilizaron para ello, y si los datos estaban o no normalizados (presencia o no del sufijo “Norm”).
2. Top *k* porcentaje de palabras: el ya explicado Top de palabras ponderadas y ordenadas decrecientemente.
3. *ICF* utilizado, de los tres disponibles.
4. Tipo de vecindad: *gaussian* en todos ellos.
5. Forma de la rejilla: hexagonales (*hexa*) en todas las configuraciones.
6. Forma del mapa: en forma de hoja (*sheet*) en la totalidad de redes. Tanto esta característica como las dos anteriores, el tipo de vecindad y la forma de la rejilla, no se pondrán en la Tabla 19 porque no varían entre las diferentes configuraciones de *SOM* testeadas.
7. Tamaño del mapa de Kohonen, cuadrados todos ellos.
8. Número de *epochs*.
9. Carpeta: para la nomenclatura (a nivel interno).
10. Número de intento: será 1 siempre que se use inicialización lineal, debiendo cambiar en caso de usar la inicialización aleatoria por requerir ésta de múltiples ejecuciones.

Tabla 19: Conjunto de soluciones finales: ranking de clasificadores según AUC (ambos escenarios).

Posición <i>ranking</i>	Escenario 1 (datos sin normalizar)				Escenario 2 (datos normalizados)			
	Top %	ICF	Tamaño	Epochs	Top %	ICF	Tamaño	Epochs
1	25	log	20x20	100	95	sqrt	20x20	8000
2	25	log	20x20	500	100	sqrt	20x20	8000
3	25	log	25x25	8000	10	linear	13x13	5000
4	25	log	25x25	100	10	linear	13x13	100
5	10	sqrt	25x25	500	10	linear	20x20	8000
6	10	log	25x25	5000	75	sqrt	20x20	100
7	10	sqrt	20x20	5000	10	linear	13x13	8000
8	10	sqrt	25x25	1000	25	sqrt	20x20	8000
9	25	sqrt	20x20	100	25	sqrt	20x20	5000
10	25	sqrt	25x25	1000	50	linear	13x13	500
11	10	log	25x25	3000	10	log	20x20	5000
12	25	log	20x20	5000	100	log	20x20	100
13	25	log	20x20	1000	95	log	20x20	100

Como ya explicamos en el apartado 1.4.3, existen dos grandes categorías de métricas que hemos implementado y utilizado: las que miden la calidad del mapa *SOM* y las que miden el rendimiento del clasificador. Con el fin de dirimir cuál es el mejor clasificador y obtener un *ranking*, hemos seleccionado al *AUC* como la métrica principal y más importante por las razones ya expuestas. Dicho *ranking* de 13 clasificadores¹¹² podemos encontrarlo en la Tabla 19, seguido tanto de la Tabla 22 a Tabla 25 (que contienen los valores de las métricas asociados con estos mejores candidatos *SOM* durante la fase de entrenamiento y ajuste) como de la Tabla 26 (con los valores de las métricas de los dos mejores clasificadores, uno para cada escenario, durante la fase de testeo).

En prime lugar, presentamos en la Tabla 20 los rangos o conjuntos de valores de las dos métricas relativas a la calidad del mapa – *MQE* y *TE* – obtenidas en algunos experimentos de otros autores. Esto serán un referente y nos ayudará en la valoración e interpretación de los resultados obtenidos en nuestros desarrollos. Si comparamos dicha tabla con las Tabla 22 a Tabla 26, nuestros valores de *MQE* y *TE* cuando se usan datos normalizados están en el mismo orden que los de la mayoría de dichos autores, de lo que se infiere una interpretación positiva de la técnica desarrollada. El hecho de que nuestro *MQE* con datos sin normalizar dé superior en varios órdenes de magnitud con respecto a los de los otros autores presumiblemente se deba a los rangos o conjuntos de valores de los datos que ellos utilizan o a que los de ellos estén normalizados.

¹¹² Debido a que se ha utilizado la función "*plot*" de MATLAB para graficar las curvas *ROC* nos hemos visto limitados a dibujarla para máximo 13 clasificadores debido a que es el máximo número de símbolos, o marcadores, diferentes que admite dicha función, los cuales son: {'+', 'o', '*', '.', 'x', 's', 'd', '^', 'v', '>', '<', 'p', 'h'}.

Tabla 20: Rangos o conjuntos de valores de MQE y TE obtenidos por otros autores.

Medición de calidad Autores	MQE	TE
(Kiviluoto 1996)	(0,04 – $\gg 0,1$) ¹¹³	(0 – 0,2)
(Pözlbauer 2004)	(0,2092 – 1,7325)	(0,0067 – 0,0551)
(Uriarte y Martín 2005)	(0,3731 – 1,2761)	(0 – 0,1525)
(Sassi 2011)	{0,251 , 4,688}	{0,039 , 0,055}

Por otra parte, hemos realizado una calificación de clasificadores, atendiendo a distintos rangos de valores que puede tomar la *AUC*, siendo esta una calificación cualitativa (Tape y University of Nebraska Medical Center 2014). Dicha calificación es bastante dependiente del ámbito de aplicación: mientras que en psicología aplicada y en predicción de comportamientos futuros, una *AUC* de 0,7 o más es muy buena, en medicina se suele ser más estricto, por los riesgos que conllevan malas clasificaciones, por lo que *AUC* superiores a 0,95 son las consideradas como muy buenas. Teniendo este hecho en consideración, hemos adecuado nuestra calificación de manera acorde (Tabla 21). Sobre los resultados en dicha tabla, destacamos que la totalidad de nuestras configuraciones de *SOM*, usando tanto datos sin normalizar como normalizados, se pueden considerar como muy buenas o excelentes, no habiendo ninguna con *AUC* menores a 0,9. Como recordaremos del apartado 1.4, los clasificadores cuyas *AUC* son menores a 0,5 pueden mejorar simplemente invirtiendo la decisión tomada, siendo los que tienen una *AUC* igual a 0,5 de igual – pésima – calidad que la decisión aleatoria. Por otro lado, el clasificador perfecto sería aquel cuyo *AUC* es igual a la unidad y que nosotros incluiríamos en el grupo de excelentes.

Tabla 21: Recuento y calificación de clasificadores en cada rango de *AUC*, para ambos escenarios.

Escenario Calificación y rangos de <i>AUC</i>	Escenario 1 (datos sin normalizar)	Escenario 2 (datos normalizados)
Pésimo → $AUC \leq 0,5$	0	0
Muy malo → $0,5 < AUC \leq 0,6$	0	0
Malo → $0,6 < AUC \leq 0,7$	0	0
Regula → $0,7 < AUC \leq 0,8$	0	0
Bueno → $0,8 < AUC \leq 0,9$	0	0
Muy bueno → $0,9 < AUC \leq 0,95$	533	523
Excelente → $AUC > 0,95$	97	107

Respecto a los resultados obtenidos para el escenario 1 (datos sin normalizar; Tabla 22 y Tabla 23) por los 13 mejores clasificadores (Tabla 19) podemos comentar lo siguiente:

- Todos los mejores clasificadores de spam/ham presentan un tamaño de 20x20 ó 25x25, lo cual indica que no se puede considerar una regla, la idea original de que a mayores tamaños de *SOM*, mejores clasificaciones se obtienen, ni la de que 13x13, dimensión acorde con el nº de categorías temáticas, sería el mejor (aunque sí obtienen buenos resultados), como

¹¹³ Se han obtenido de gráficas (ya que el autor no proporciona los valores exactos) por lo que no son del todo precisos.

recomendaba (Kohonen 2001, 2013).

- Todas las 13 configuraciones analizadas como conjunto de posibles soluciones finales, obtienen excelentes resultados en *AUC*, similares entre sí, tal y como se podía inferir del recuento en la Tabla 21.
- Tanto la especificidad como la sensibilidad, muestran valores bastante buenos y próximos entre todas las configuraciones, teniendo valores menores los de la primera de estas métricas.
- F-score, “*accuracy*” y “*precision*”, siendo menores los de esta última métrica, cuentan con valores igualmente positivos.
- La medida de la distancia al punto (0, 1) desde la curva *ROC*, que indicamos como posible métrica, también presenta valores similares en dichas configuraciones de *SOM* aunque no podemos inferir una correspondencia clara ni directa entre menor distancia y mejor posición en el *ranking*.
- Tanto los *FP* como los *FN* suelen rondar el 10% y 9,8% (2,65% en el clasificador 4, mejor caso) de los *TP* y *TN*, respectivamente, lo cual deberá ser mejorado en futuros trabajos relacionados. Este resultado era esperable dada la dificultad que entraña la existencia de correos grises, los cuales habría que tratar con especial cuidado, posiblemente durante el preprocesado.
- Todos los mejores clasificadores utilizan Top 10% o Top 25% (Tabla 19), es decir, usan los menores porcentajes de palabras por lo que disponer de mayor cantidad de palabras no ha mejorado el poder de clasificación del *SOM*.
- El *ICF* linear, pese a tener resultados de *AUC* similares aunque algo inferiores a los mejor posicionados, no aparece en el *ranking* hasta la posición 25 (el *ranking* recogido en la Tabla 19 sólo llega hasta la posición 13). Ello puede indicarnos que las opciones log y sqrt son las mejores con datos sin normalizar.

Observando los clasificadores que obtuvieron peores posiciones (por ello no tabulados aquí), destaca que los que cuentan con tamaño 50x50, los de mayor tamaño de los probados, e *ICF* linear, poseen *AUC* peores, aún así manteniéndose dentro del rango de *AUC* que permite calificarlos como muy buenos.

Tabla 22: Métricas obtenidas para los 1-6 mejores clasificadores del escenario 1 (Fase de Entrenamiento-Ajuste).

Posición del clasificador Métricas	1	2	3	4	5	6
<i>AUC</i>	0,963185	0,960386	0,959660	0,959594	0,959408	0,958994
<i>MQE</i>	51,343423	50,812503	47,264532	47,899662	35,431636	39,016577
<i>TE</i>	0,124054	0,116994	0,081190	0,083207	0,106404	0,075139
<i>AUC mejor discreto relacionado & “Threshold”</i>	0,915953 51	0,907759 58	0,920894 45	0,912937 14	0,914443 41	0,912531 43
<i>Distancia al punto (0, 1)</i>	0,119937	0,131263	0,116816	0,154360	0,125059	0,127648
<i>Especificidad</i>	0,904615	0,897436	0,897119	0,847107	0,892086	0,890256
<i>Sensibilidad</i>	0,927291	0,918082	0,944668	0,978766	0,936799	0,934805
<i>F-score</i>	0,918146	0,909901	0,923758	0,919715	0,916708	0,913200
<i>“Accuracy”</i>	0,916119	0,907895	0,921160	0,913643	0,914534	0,912253
<i>“Precision”</i>	0,909180	0,901865	0,903754	0,867384	0,897461	0,892570
<i>TP</i>	931	919	939	968	919	889
<i>FP</i>	93	100	100	148	105	107
<i>TN</i>	882	875	872	820	868	868
<i>FN</i>	73	55	55	21	62	62

Tabla 23: Métricas obtenidas para los 7-13 mejores clasificadores del escenario 1 (Fase de Entrenamiento-Ajuste).

Posición del clasificador Métricas	7	8	9	10	11	12	13
<i>AUC</i>	0,957621	0,957446	0,957337	0,957296	0,957219	0,956873	0,956711
<i>MQE</i>	37,980127	35,346816	46,562389	43,270800	39,069729	50,760081	50,738147
<i>TE</i>	0,187090	0,115482	0,136662	0,108422	0,079677	0,124054	0,116994
<i>AUC mejor discreto relacionado & “Threshold”</i>	0,909763 59	0,915421 34	0,915502 37	0,918264 37	0,913003 39	0,907675 58	0,904574 58
<i>Distancia al punto (0, 1)</i>	0,129769	0,122445	0,132595	0,123273	0,127543	0,130791	0,135622
<i>Especificidad</i>	0,893114	0,896907	0,874872	0,887975	0,889231	0,902263	0,895062
<i>Sensibilidad</i>	0,926411	0,933934	0,956132	0,948553	0,936776	0,913087	0,914086
<i>F-score</i>	0,912159	0,918307	0,920345	0,918526	0,913669	0,909453	0,906838
<i>“Accuracy”</i>	0,909924	0,915693	0,916077	0,917629	0,912682	0,907755	0,904714
<i>“Precision”</i>	0,898338	0,903195	0,887142	0,890342	0,891675	0,905847	0,899705
<i>TP</i>	919	933	959	885	889	914	915
<i>FP</i>	104	100	122	109	108	95	102
<i>TN</i>	869	870	853	864	867	877	870
<i>FN</i>	73	66	44	48	60	87	86

Respecto a los resultados obtenidos para el escenario 2 (con vectores de datos normalizados; Tabla 24 y Tabla 25) por los 13 mejores clasificadores (Tabla 19) podemos aducir lo siguiente:

- Todos los 13 mejores clasificadores presentan un tamaño de 13x13 ó 20x20, lo cual entra dentro de una de las recomendaciones de (Kohonen 2001, 2013). Al igual que con datos sin normalizar, con mayores tamaños de mapa no se obtuvieron mejores puestos en el *ranking* (aunque los valores de las métricas son parecidos) por lo que probar con mayores aún puede incluso ser contraproducente.
- Dichas 13 configuraciones obtienen buenos resultados en *AUC*, bastante similares en estos puestos de la tabla clasificatoria. Son ligeramente inferiores a los obtenidos cuando no se aplica ningún tipo de normalización (escenario 1).
- En el resto de métricas ocurre se encuentra una situación parecida a la del escenario 1 (Tabla 22 y Tabla 23). Debemos volver a indicar que los valores de *FP* y *FN* rondan el 10% (*FN* 5,88% en el clasificador 11, mejor caso) de los *TP* y *TN* en la matriz de confusión, algo que será deseable disminuir en el futuro.
- Los resultados de Top 100% y Top 95% obtienen los mismos resultados (clasificadores 1 con 2 y 12 con 13, Tabla 19) de ahí que, en dicho caso, disponer de la máxima cantidad de palabras no ha mejorado el poder de clasificación del SOM. En este caso, hay bastante variedad de Top, no pudiendo nosotros concretar cuál es el mejor en este escenario.
- Al contrario que ocurrió anteriormente, el *ICF* linear no implica peores posiciones. Por ello, y al contrario que pudimos concluir en el escenario 1, no podemos afirmar fehacientemente

cuál de las opciones de *ICF* es la más idónea con datos normalizados.

Tabla 24: Métricas obtenidas para los 1-6 mejores clasificadores del escenario 2 (Fase de Entrenamiento-Ajuste).

Posición del clasificador Métricas	1	2	3	4	5	6
<i>AUC</i>	0,956554	0,956554	0,956266	0,955968	0,955516	0,955510
<i>MQE</i>	0,544872	0,544872	0,667073	0,670844	0,592957	0,544498
<i>TE</i>	0,073122	0,073122	0,052950	0,061019	0,061523	0,061019
<i>AUC mejor discreto relacionado & “Threshold”</i>	0,910044 37	0,910044 37	0,900426 61	0,898669 57	0,909830 43	0,911140 34
<i>Distancia al punto (0, 1)</i>	0,137963	0,137963	0,142739	0,143327	0,131522	0,133218
<i>Especificidad</i>	0,872297	0,872297	0,916923	0,900513	0,887064	0,879877
<i>Sensibilidad</i>	0,947791	0,947791	0,883929	0,896825	0,932596	0,942403
<i>F-score</i>	0,914729	0,914729	0,900000	0,899950	0,912851	0,915581
<i>“Accuracy”</i>	0,910524	0,910524	0,900151	0,898638	0,910061	0,911661
<i>“Precision”</i>	0,883895	0,883895	0,916667	0,903097	0,893925	0,890244
<i>TP</i>	944	944	891	904	927	949
<i>FP</i>	124	124	81	97	110	117
<i>TN</i>	847	847	894	878	864	857
<i>FN</i>	52	52	117	104	67	58

Tabla 25: Métricas obtenidas para los 7-13 mejores clasificadores del escenario 2 (Fase de Entrenamiento-Ajuste).

Posición del clasificador Métricas	7	8	9	10	11	12	13
<i>AUC</i>	0,954691	0,954633	0,954489	0,954437	0,954427	0,954408	0,954408
<i>MQE</i>	0,667085	0,497919	0,497813	0,742512	0,477013	0,560611	0,560611
<i>TE</i>	0,054967	0,069592	0,070600	0,047403	0,065053	0,052950	0,052950
<i>AUC mejor discreto relacionado & “Threshold”</i>	0,900426	0,915388	0,915432	0,910818	0,911022	0,911691	0,911691
	61	61	61	63	31	39	39
<i>Distancia al punto (0, 1)</i>	0,142739	0,119779	0,119601	0,131251	0,137520	0,129653	0,129653
<i>Especificidad</i>	0,916923	0,911614	0,914697	0,885128	0,871795	0,887064	0,887064
<i>Sensibilidad</i>	0,883929	0,919162	0,916168	0,936508	0,950249	0,936318	0,936318
<i>F-score</i>	0,900000	0,916874	0,916625	0,914729	0,916067	0,915370	0,915370
<i>“Accuracy”</i>	0,900151	0,915543	0,915443	0,911246	0,911616	0,912077	0,912077
<i>“Precision”</i>	0,916667	0,914598	0,917083	0,893939	0,884259	0,895338	0,895338
<i>TP</i>	891	921	918	944	955	941	941
<i>FP</i>	81	86	83	112	125	110	110
<i>TN</i>	894	887	890	863	850	864	864
<i>FN</i>	117	81	84	64	50	64	64

Para la fase de testeo, los resultados (Tabla 26) demuestran que los dos mejores clasificadores, uno por cada escenario, han sido capaces de clasificar datos completamente desconocidos con la diligencia que se esperaba. Se han obtenido mejores resultados con todas las métricas cuando se utilizaron datos normalizados durante la fase de aprendizaje, no habiendo grandes diferencias cuando no se aplicó normalización.

En base a todo lo expuesto, el mejor clasificador, el cual sería el finalmente elegido para utilizarlo en ulteriores pruebas y/o en entornos reales, de los 1260 aquí probados es “BmusTrainTestNorm Top 95 porc - ICF sqrt - gaussian hexa sheet 20x20 8000 epochs mini mix – 1” (candidato número 1 en la Tabla 19), galardón que deberá compartir con “BmusTrainTestNorm Top 100 porc - ICF sqrt - gaussian hexa sheet 20x20 8000 epochs mini mix – 1” (candidato número 2 en la Tabla 19) puesto que obtienen los mismos resultados y sólo se diferencian en la cantidad de palabras ponderadas que se toman de cada categoría: el 95% o el 100%, respectivamente. Ergo y teniendo en cuenta que sólo uno puede ser el finalista, podemos finalmente concluir que el mejor clasificador es el del Top 95% debido a que es más rápido ya que consigue idénticos resultados haciendo uso de un menor número de palabras.

Tabla 26: Métricas obtenidas para los mejores clasificadores de ambos escenarios (Fase de Testeo).

Escenario Métricas	Escenario 1 (datos sin normalizar)	Escenario 2 (datos normalizados)
<i>AUC</i>	0,970809	0,977740
<i>MQE</i>	39,592070	0,452661
<i>TE</i>	0,118567	0,078204
AUC mejor discreto relacionado & “Threshold”	0,924172 38	0,944499 37
Distancia al punto (0, 1)	0,118007	0,083623
Especificidad	0,889344	0,924103
Sensibilidad	0,959000	0,964895
<i>F-score</i>	0,927915	0,946385
<i>“Accuracy”</i>	0,924595	0,944726
<i>“Precision”</i>	0,898782	0,928571
<i>TP</i>	959	962
<i>FP</i>	108	74
<i>TN</i>	868	901
<i>FN</i>	41	35

A continuación, presentamos las curvas ROC (desde la Figura 21 hasta la Figura 29) pertenecientes a los clasificadores cuyos valores hemos tabulado y comentado anteriormente. Cada una de dichas configuraciones obtienen curvas similares entre sí, de ahí que se confirma que no existan grandes diferencias en los resultados tabulados. Vistas dichas curvas ROC podemos afirmar que cada uno de los mejores clasificadores para cada uno de los dos escenarios (datos sin normalizar y datos normalizados), tanto durante la fase de entrenamiento-ajuste (Figura 21 a Figura 28) como durante la de testeo (Figura 29), se caracterizan por ser de excelente calidad (como nos indicaba la Tabla 20) por varios motivos:

- Pasan cerca del punto de clasificación perfecta en (0, 1), lo que en las anteriores tablas se deduce observando los valores bastante altos de especificidad y sensibilidad.
- Tienen bastante poder de discriminación, debido a estar bastante alejados de la línea de decisión aleatoria (discontinua y en color rojo en todas las figuras).
- Los valores de *AUC* asociados a dichas curvas ROC son bastante altos debido a los anteriores puntos, quedando demostrado lo expuesto en las tablas precedentes.

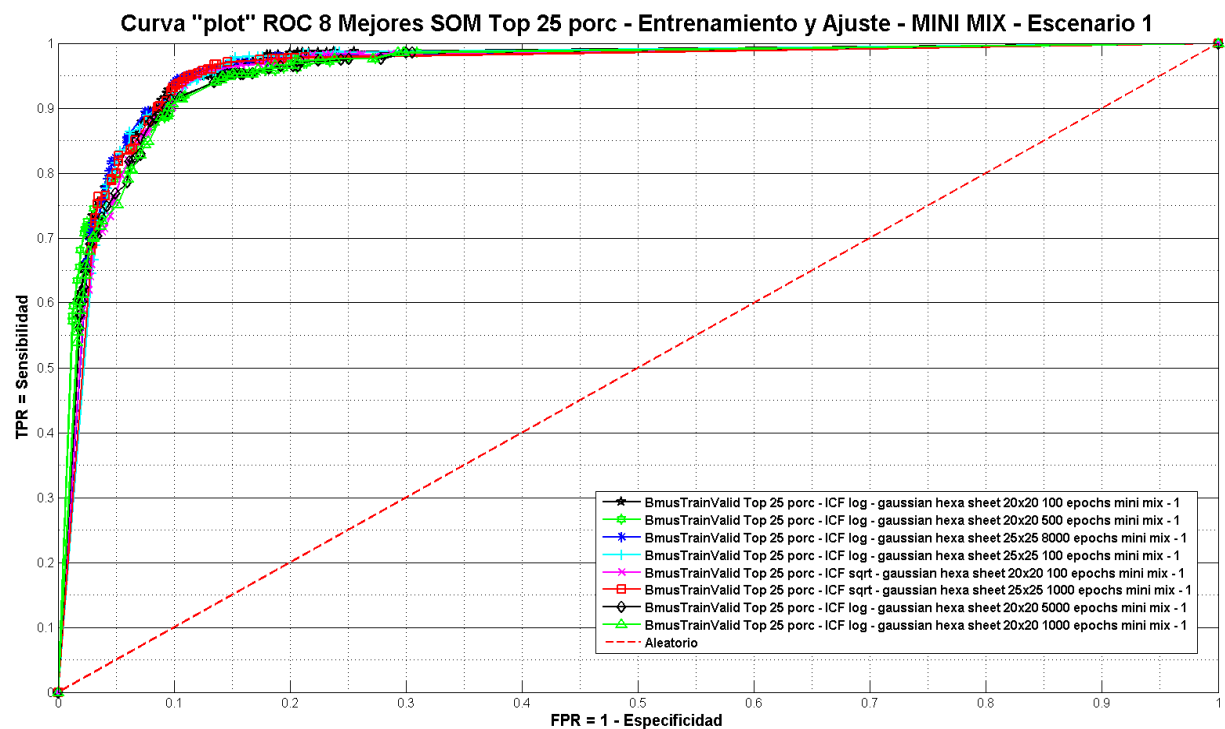


Figura 21: Curvas ROC de 8 mejores SOM Top 25 – Escenario 1 – Fase de Entrenamiento-Ajuste

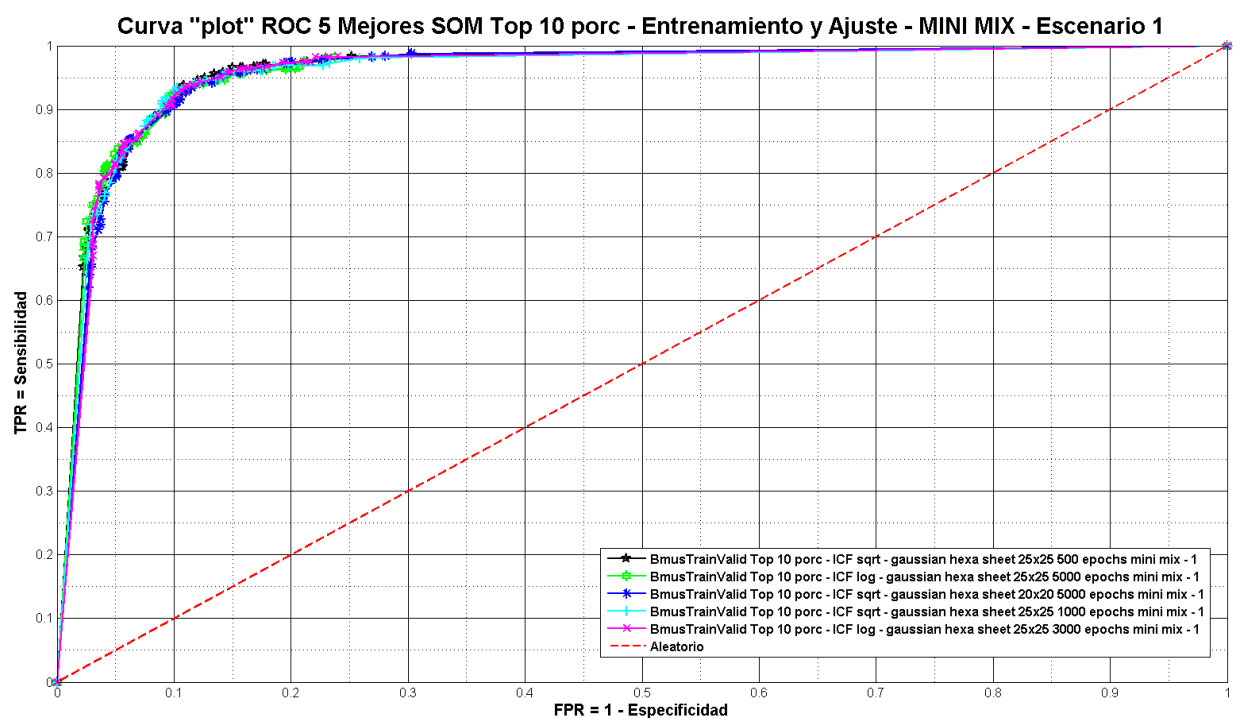


Figura 22: Curvas ROC de 5 mejores SOM Top 10 – Escenario 1 – Fase de Entrenamiento-Ajuste

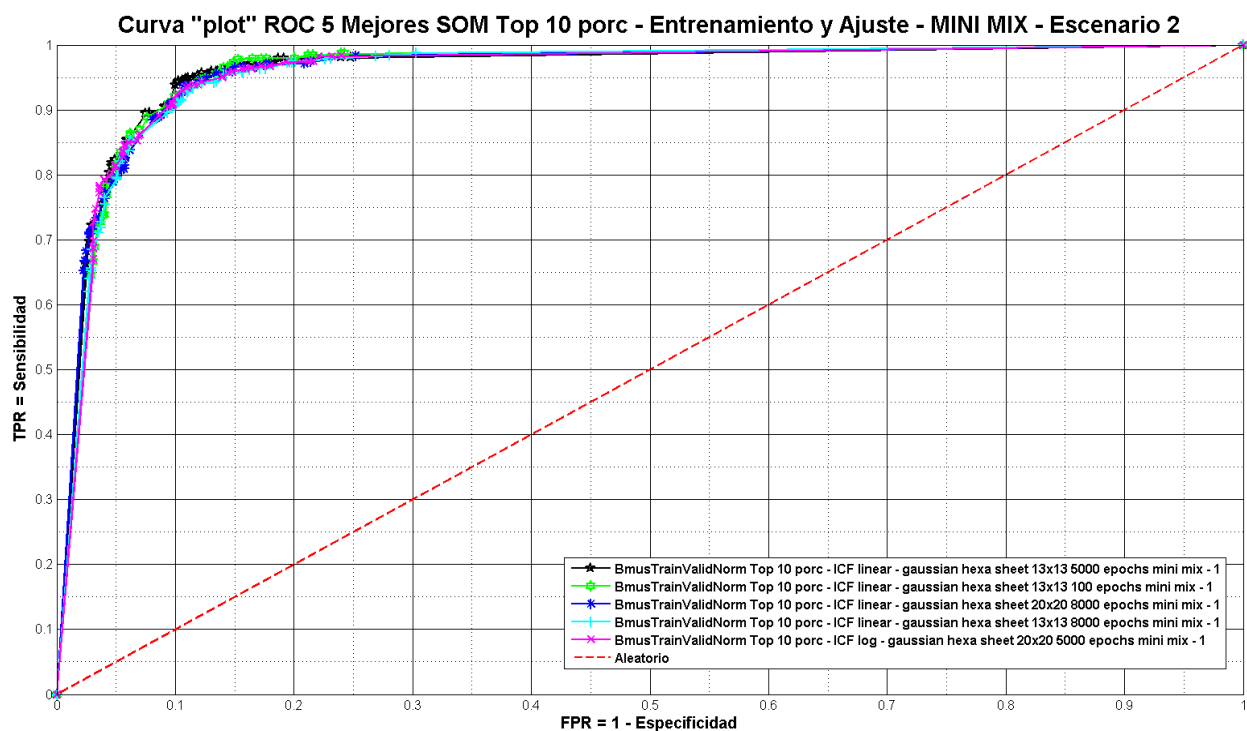


Figura 23: Curvas ROC de 5 mejores SOM Top 10 – Escenario 2 – Fase de Entrenamiento-Ajuste

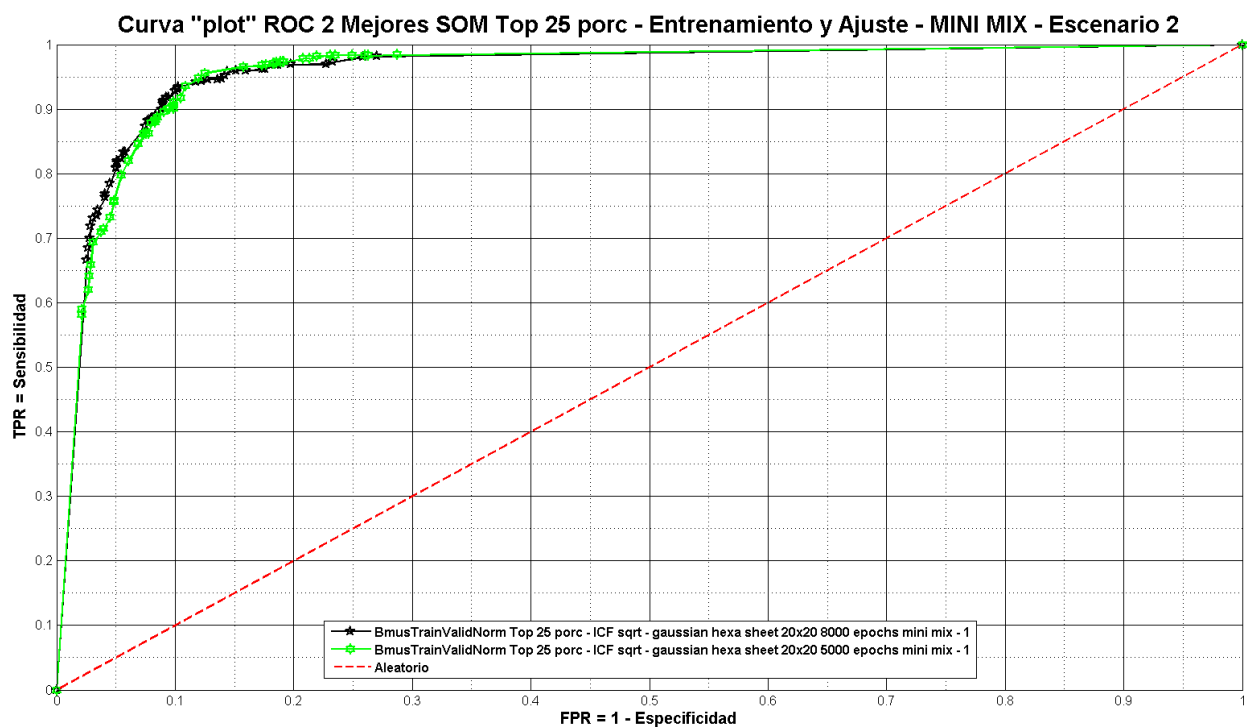


Figura 24: Curvas ROC de 2 mejores SOM Top 25 – Escenario 2 – Fase de Entrenamiento-Ajuste

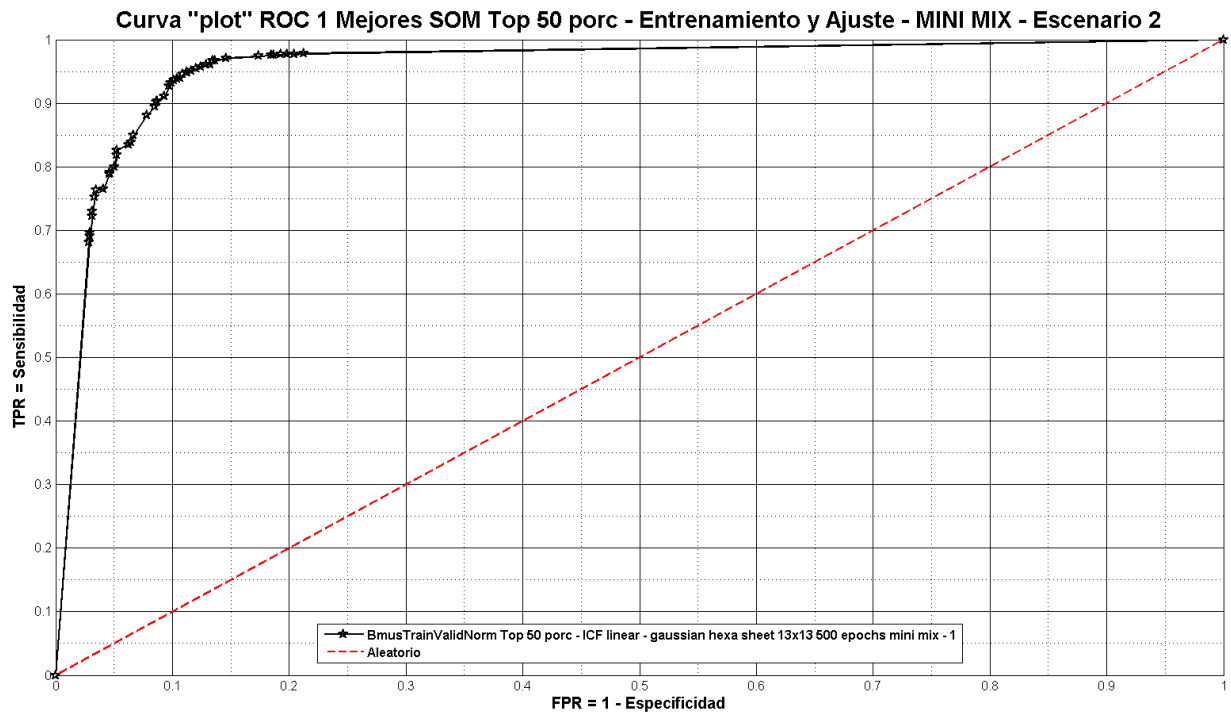


Figura 25: Curva ROC del mejor SOM Top 50 – Escenario 2 – Fase de Entrenamiento-Ajuste

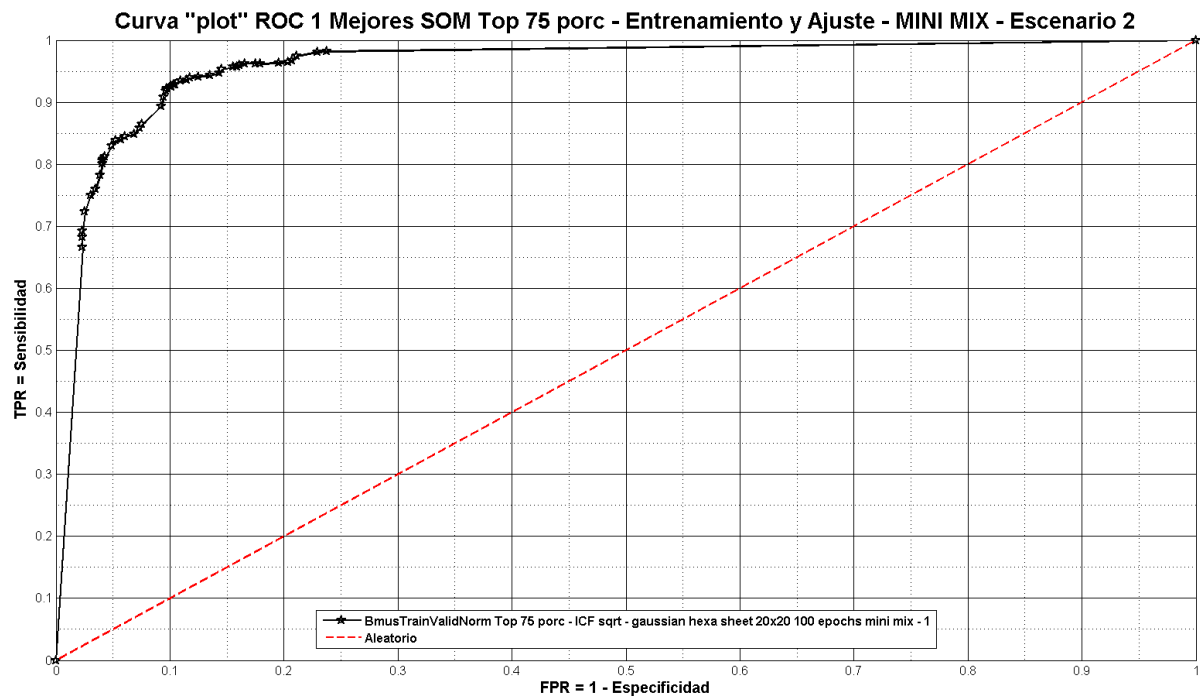


Figura 26: Curva ROC del mejor SOM Top 75 – Escenario 2 – Fase de Entrenamiento-Ajuste

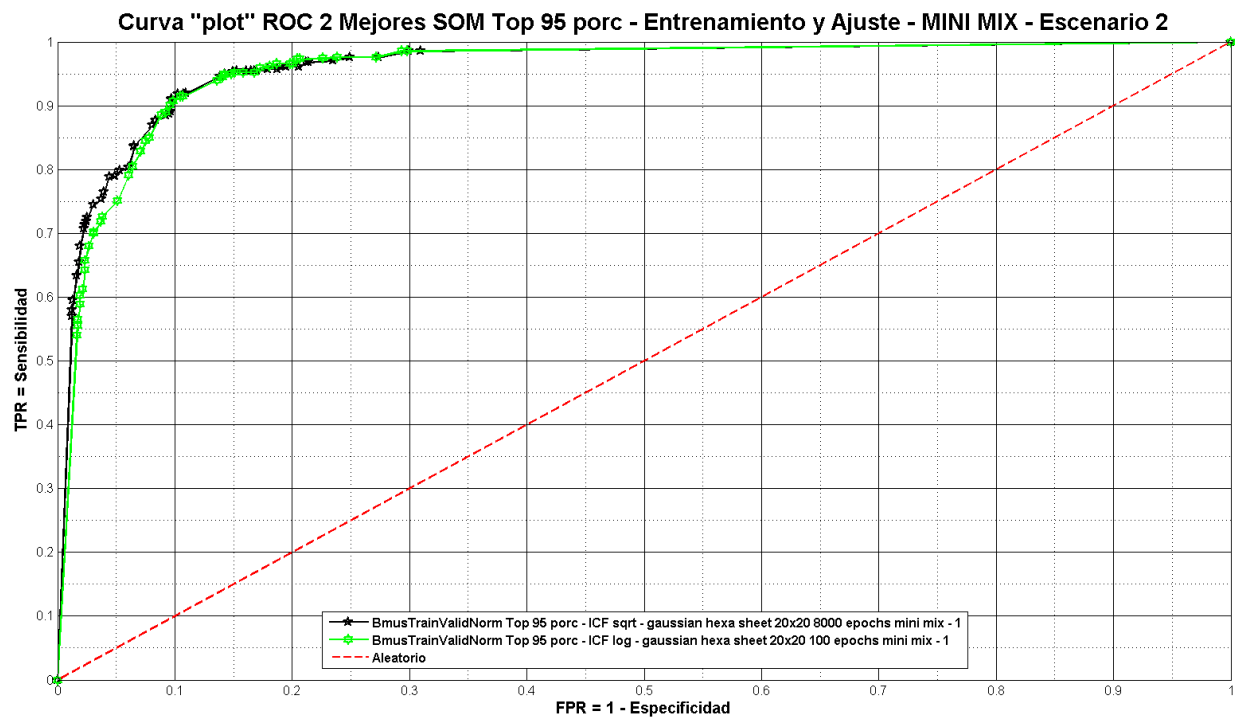


Figura 27: Curvas ROC de 2 mejores SOM Top 95 – Escenario 2 – Fase de Entrenamiento-Ajuste

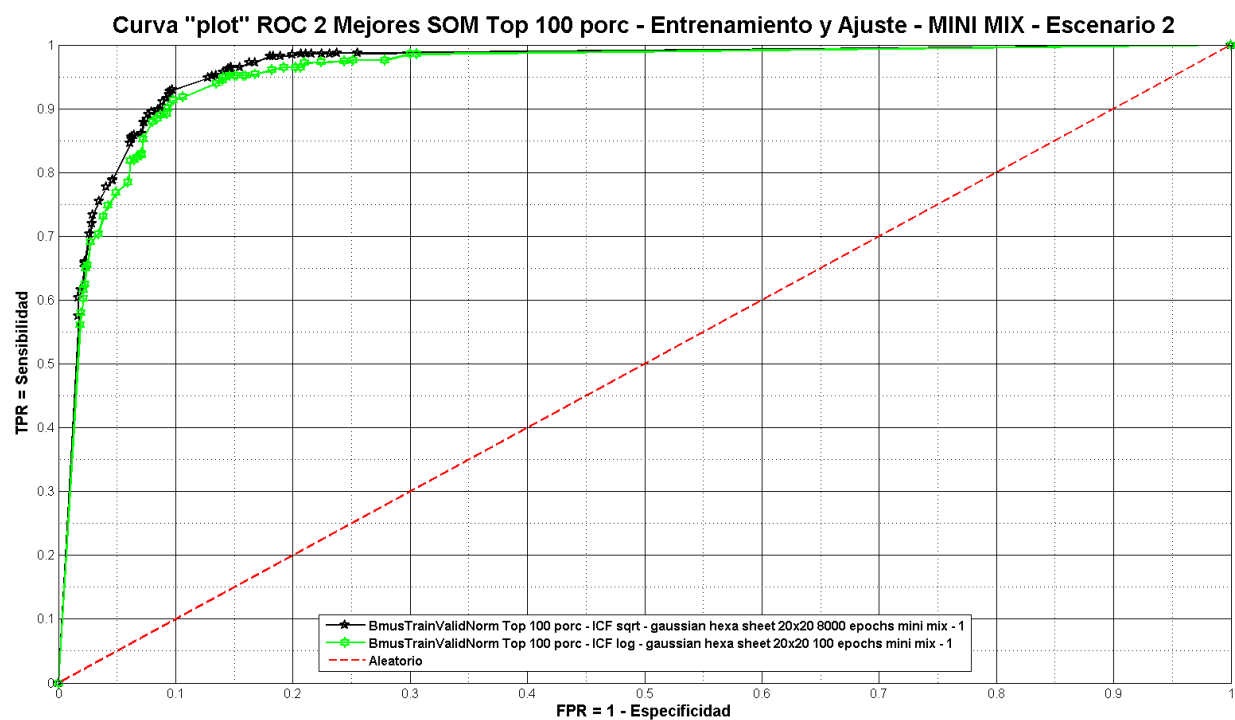


Figura 28: Curvas ROC de 2 mejores SOM Top 100 – Escenario 2 – Fase de Entrenamiento-Ajuste

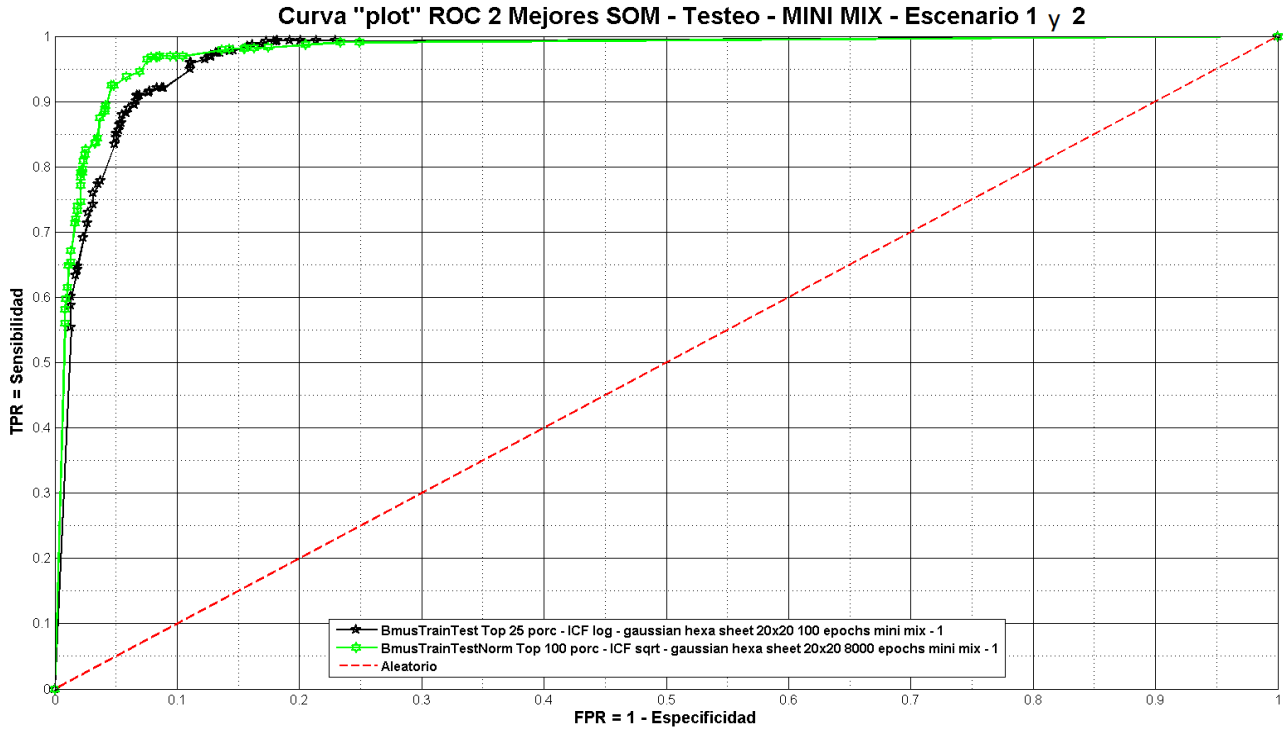


Figura 29: Curvas ROC de los 2 mejores SOM – Ambos escenarios – Fase de Testeo

Algo similar ha ocurrido cuando se han graficado las curvas ROC de los dos mejores SOM sobre el conjunto de testeo (Figura 29), obteniéndose mejores resultados cuando se usa normalización, algo ya observado en la Tabla 26: pasan cerca del punto de clasificación perfecta en (0, 1) al tiempo que bastante alejado de la línea de decisión aleatoria.

3.3.2.1 Coste temporal

En la Tabla 27 viene detallado el coste temporal aproximado, en ejecución ininterrumpida, de algunas de las funciones implementadas, muchos de los cuales son convenientemente mostrados al usuario en tiempo de ejecución. Dicha tabla contiene, también, el nombre de los ficheros en los cuales vienen definidas dichas funciones y sus respectivas funcionalidades. Cabe recordar que todos ellos se refieren a la ejecución sobre el hardware y software indicados en el Capítulo 2. Es de esperar que estos tiempos se puedan ver reducidos todos ellos, incluso en varios órdenes de magnitud, si la ejecución hubiera sido realizada en sistemas más potentes y paralelizados.

Como se puede observar en la Tabla 27, el coste temporal ha sido bastante alto, primordialmente tanto durante el cálculo de los valores de la matriz de confusión (*i.e.* TP , FP , TN y FN) para la fase de entrenamiento-ajuste como, en menor medida, durante la etapa de aprendizaje de las diferentes configuraciones de SOM. La razón de que el primero de ellos haya sido tan elevado, superando 12 días, nos ha sorprendido y pensamos que se ha debido a que, para cada una de las 1260 configuraciones de SOM, se han tenido que calcular los valores de TP , FP , TN y FN para cada uno de los 102 *thresholds* necesarios para graficar las curvas ROC, actualizándose estas variables con cada uno de los 1983 emails del subconjunto de validación. Además, todo ello venía acompañado de operaciones de entrada-salida (tanto para recuperar información previamente almacenada como para mantener copias de seguridad de los datos nuevos obtenidos), y de

impresiones por pantalla de algunos de ellos, de carácter informativo. Nos consta que ambos tiempos crecen con el tamaño y la cantidad de mapas de Kohonen estudiados.

Tabla 27: Coste temporal y funcionalidades de algunas de las funciones implementadas.

Nombre de la función/fichero	Funcionalidad	Tiempo empleado (aprox.)
“pfc_aprender_mini_mix.m”	Aprendizaje de las diversas configuraciones de <i>SOM</i> probadas	5d 6h
“pfc_recuperar_info_aprendizaje_crear_cmapas_mini_mix.m”	Guardar en <i>container maps</i> las estructuras de datos, mapas, vectores de entrada y etiquetas usadas durante el aprendizaje	1h 35min
“pfc_valorar_clasificadores_mini_mix.m”	Cálculo de los <i>TP</i> , <i>FP</i> , <i>TN</i> y <i>FN</i> para cada clasificador y “ <i>threshold</i> ” para el conjunto de entrenamiento-ajuste	12d 10h
“pfc_guardar_info_valoracion_cmaps_TV_mini_mix.m”	Cálculo de las diferentes métricas implementadas para el conjunto de entrenamiento-ajuste + Ordenado (ascendente o descendente) de las clasificaciones según las diferentes métricas	43min 19s + 6s
“pfc_analizar_clasificadores_mini_mix.m”	Análisis de los clasificadores para el conjunto de entrenamiento-ajuste, retornando una lista del mejor según <i>AUC</i> para cada escenario, junto a todas sus métricas correspondientes. Además, se grafican las curvas <i>ROC</i>	1h 48min
“pfc_valorar_clasif_guardar_info_cmaps_T_mini_mix.m”	Para el conjunto de testeo, cálculo de los <i>TP</i> , <i>FP</i> , <i>TN</i> y <i>FN</i> para los mejores clasificadores y “ <i>threshold</i> ”. También, cálculo de las diferentes métricas implementadas	48min 29s
“pfc_analizar_clasificadores_mini_mix.m”	Análisis de los clasificadores para el conjunto de testeo, graficando curva <i>ROC</i>	4s

Capítulo 4. Conclusiones y trabajos futuros

Introducción: En este capítulo se describen las conclusiones del proyecto llevado a cabo y algunos de los posibles trabajos futuros por donde se podrá continuar investigando.

4.1 Conclusiones

De forma general podemos concluir que las principales contribuciones de este proyecto están íntimamente ligadas a sus propios objetivos, los cuales han sido alcanzados a través de los distintos estudios y desarrollos realizados a lo largo de este trabajo. Este proyecto ha permitido construir un sistema de filtrado antispam basado en RNA no supervisadas, los mapas autoorganizados de Kohonen, alcanzándose conclusiones relevantes en el tema abordado: detección automática de correos spam.

Seguidamente se exponen las principales conclusiones del proyecto:

1. Se ha diseñado un sistema antispam inteligente basado en RNA no supervisadas, más concretamente en los mapas autoorganizados de Kohonen (*SOM*). La finalidad principal de este sistema es la de distinguir los mensajes de correo electrónico no deseados (spam) del correo legítimo (ham) o, dicho de otra forma, clasificar un mensaje en alguna de esas dos categorías.
2. Se ha seleccionado un repositorio de correos electrónicos libre y gratuito, a partir del cual se generó un cuerpo de datos representativo y balanceado, con una ratio de spam y ham de 1:1.
3. Se ha demostrado que existen categorías temáticas en los spam, a través de las cuales pueden ser clasificados y diferenciados de los ham, confirmando lo expuesto por otros investigadores. Nuestra propuesta se ha basado en el uso de 13 categorías temáticas de palabras: Sexo & Relaciones, Medicina & Drogas & Cuerpo, Apuestas & Azar & Adivinación, Banca & Inversiones & Seguros & Compra-Venta, Enlace & Correo electrónico, Otros idiomas, Ofuscación; Negocios & Empresa & Gobierno, Internet & Tecnología; Documentos & Trámites, Nombres & Familia, Turismo & Regiones y Fichero adjunto.
4. Se ha desarrollado una representación de información vectorial, obtenida a través del preprocesado de los datos, desarrollado en este proyecto. Cada vector de características representa un correo electrónico del conjunto de datos y está formado por 13 atributos en coma flotante (tantos como número de categorías) y 2 etiquetas alfanuméricas (ham/spam y nombre identificador del mensaje).
5. Se ha desarrollado un nuevo método híbrido de preprocesado de los datos del espacio de entrada. Este preprocesado híbrido consta de una etapa manual y de otra automática, y está dividido en tres fases: las fases 1 y 2 están cubiertas por la etapa manual, y la fase 3, por la computacional. Con el preprocesado manual se ha llevado a cabo una categorización temática de las palabras en 13 categorías, teniendo en cuenta el contexto y uso. La etapa automática posterior pondera cada palabra dentro de cierta categoría según su importancia en esa categoría.
6. El preprocesado manual propuesto presenta dos diferencias importantes frente a los métodos de preprocesado comúnmente utilizados:
 - a) No precisa de un proceso de lematización de las palabras, lo cual es bastante usual en otros preprocesados.
 - b) Su robustez frente a intentos de ofuscación de las palabras por parte de los *spammers*.

La ofuscación es algo muy frecuente en el ámbito de los spam, como se ha demostrado, de ahí la existencia de una categoría temática de palabras que representa dicha contingencia.

7. Se ha desarrollado un método de ponderación de las palabras que conforman el conjunto representativo de correos electrónicos, con el fin de ignorar tanto las palabras demasiado comunes (*stop words*) como las poco frecuentes. Dicho método está basado en $TF*ICF$. Se ha probado con tres variantes de *ICF* propuestas por otros autores.
8. Para evaluar la bondad y el rendimiento del sistema inteligente antispam, se han utilizado varias familias de métricas: las que miden la calidad del mapa *SOM* desarrollado (*MQE* y *TE*), y las que miden el rendimiento del filtro antispam propuesto (*TP*, *FP*, *TN*, *FN*, especificidad, sensibilidad, *F-score*, “*accuracy*”, “*precision*”, curva *ROC* y *AUC*). Esta última métrica es la considerada más significativa y global para medir la bondad de un clasificador y se ha hecho uso de ella para realizar la discusión y soportar la ordenación de las distintas configuraciones de clasificadores diseñadas.
9. Se ha considerado una medida sencilla, la distancia, usando distancia euclidiana, entre el punto (0, 1) de clasificación perfecta y la curva *ROC*, para determinar la situación del mejor clasificador. Éste sería el que se encuentre situado en el punto de menor distancia al (0, 1), es decir, el que presente el mínimo valor de esta medida. Los resultados obtenidos con esta nueva métrica indican que, pese a su simpleza, puede ser de utilidad, sobre todo si se desea una con reducido coste computacional.
10. Se ha analizado la bondad e idoneidad de trabajar con datos normalizados. Los resultados alcanzados para el filtro inteligente antispam son óptimos cuando se trabaja con el espacio de entrada normalizado, lo que se ajusta a las características de funcionamiento de los mapas autoorganizados de Kohonen.
11. Se han estudiado 1260 configuraciones distintas, las cuales se diferencian entre sí en el tamaño del *SOM*, en la normalización de los datos de entrada, en el número de *epochs* durante los cuales se ha entrenado la red, en el *ICF* utilizado y, por último, en el porcentaje de palabras ponderadas que mayor importancia tuvieran. Los valores para el resto de características se han elegido de acuerdo a las recomendaciones de otros autores, entre ellos Teuvo Kohonen, creador del *SOM*.
12. El porcentaje de palabras ponderadas ejerce influencia en la calidad del filtro antispam. Se ha observado que, para el escenario 1, valores menores, tales como Top 10% y Top 25%, presentaron mejores resultados. En ambos escenarios las configuraciones que han hecho uso de la totalidad del conjunto de palabras de cada categoría (Top 100%) han obtenido exactamente los mismos resultados que las mismas configuraciones que sólo hicieron uso del Top 95%. Se puede concluir que disponer de la máxima cantidad de palabras no mejora el poder de detección del sistema, siendo recomendable el uso del Top 95%, lo que además proporcionaría mayor rapidez.
13. Todas las configuraciones obtienen resultados bastante similares entre ellas en las diferentes métricas implementadas, en especial si comparamos configuraciones que compartan escenario. Dichos resultados han sido bastante positivos. Los valores menos favorables han estado centrados en los *FP* y *FN*, hasta del 10% en los peores casos, los cuales sería deseable disminuir.

14. Las 1260 configuraciones probadas de nuestro sistema de filtrado antispam obtienen resultados entre muy buenos ($0,9 < AUC \leq 0,95$) y excelentes ($AUC > 0,95$), no habiendo ninguna que obtuviera $AUC < 0,9$. Para el escenario 1 se obtuvieron 533 muy buenas y 97 excelentes mientras que para el escenario 2 fueron 523 y 107, respectivamente.
15. Las 13 mejores configuraciones, o conjunto de posibles soluciones finales, tienen buen poder de discriminación y las decisiones tomadas por los clasificadores pueden ser consideradas correctas. Se observa que, en conjunto, presentan menores valores de FP y FN , y más TP y TN . Son las configuraciones que alcanzan los mejores valores para esos dos pares de métricas al mismo tiempo, que es lo que las diferencia de las restantes.
16. Para el escenario 1 se puede inferir que ICF linear no es el más idóneo aunque no se puede concluir fehacientemente lo mismo para el escenario 2.
17. Se observa que mayores tamaños de SOM no implican mejores resultados para el filtro antispam.
18. La opción más prometedora de todas las 1260 configuraciones resulta ser una probada sobre el escenario 2, es decir, a los datos de entrada se les aplicó un método de normalización. Obtiene mejores resultados en todas las métricas utilizadas que la mejor configuración para el escenario 1, con datos sin normalizar.

4.2 Trabajos futuros

Igualmente, se nos han abierto algunas posibles futuras líneas de investigación y de trabajo, entre las que podemos mencionar las siguientes:

1. Implementación en ciertos entornos de desarrollo (tales como GNU Octave, Weka o R) o con lenguajes de programación de uso general (como Python, Java, Ruby y C++¹¹⁴), en ambos casos libres y gratuitos. Aparte de ahorrar costes, también se pretendería aumentar el trabajo colaborativo o hacer más eficiente nuestro código.
2. Mejorar la eficiencia y el rendimiento de la implementación (*e.g.* incrementar el nivel de paralelización del código, reducir el consumo de memoria tanto primaria como secundaria...). Realizar la implementación del sistema propuesto en infraestructuras de cómputo de alto rendimiento, tales como el clúster de altas prestaciones COMCIENCIA o el supercomputador Atlante, con el fin de reducir los tiempos de ejecución.
3. Ampliación de las capacidades del software desarrollado agregando nuevas funcionalidades (Ej.: mejorar la interfaz gráfica de usuario, conseguir mayor robustez durante la E/S, etc).
4. Probar con otras configuraciones de SOM (*i.e.* podría ser de gran interés hacerlo con otros tamaños y formas de SOM , con inicialización aleatoria o con el algoritmo secuencial).

¹¹⁴ En estos lenguajes de programación está disponible la implementación de un diccionario, tabla de *hash*, mapa o multimapa (en sus versiones ordenadas o sin ordenar) que pueden ser equivalentes al *container map* disponible en MATLAB y que ha sido bastante utilizado en este proyecto.

5. Testear más profusamente la idoneidad y potencialidad de la nueva métrica del poder de clasificación desarrollada, probablemente haciendo una versión mejorada más potente y versátil. Para ello, nuestra nueva métrica mejorada podría tener en cuenta no sólo la menor distancia sino otra clase de información posiblemente útil como, por ejemplo, el ángulo entre la recta formada por el punto (FPR , TPR) más cercano de la curva ROC para dicho clasificador y el punto (0, 1) de clasificación perfecta, y el eje X (o de FPR).
6. Trabajar el esquema de preprocesado. Desarrollar nuevos modelos y métodos de preprocesado así como analizar más variantes propuestas por otros autores.
7. Analizar otras métricas de calidad de los clasificadores diferentes.
8. No usar el método *bag-of-words* durante la fase 1 del preprocesado sino algún otro que sí tenga en cuenta el orden de las palabras (Ej.: n-gramas).
9. Trabajar con algún cuerpo de ham más genérico que el aquí utilizado¹¹⁵ y, si es posible, con correos electrónicos más actuales.
10. Investigar con otras RNA, tanto con aprendizaje no supervisado (*e.g. neural gas*) como supervisado o, incluso, híbrido como la red *counterpropagation* (CPN) y la arquitectura *HUMANN*.
11. Desarrollar un filtro antispam multiclase¹¹⁶. Este consistiría en un sistema inteligente de clasificación multiclase con capacidad para detectar casos dudosos o grises.
12. Desarrollo de sistemas de clasificación automática de correos electrónicos en carpetas temáticas, sistemas inteligentes con capacidad tanto de clasificación multiclase como de clasificación multietiqueta¹¹⁷.

115 *Per se* proveniente de usuarios de la empresa energética Enron *ergo* el contenido de los ham es bastante peculiar y particular, no necesariamente extrapolable a la mayoría de los usuarios de correo electrónico.

116 En este caso, clasificar cada mensaje en una de las más de dos carpetas temáticas (clases) posibles. Cuando sólo existen dos clases posibles, se trata de clasificación binaria.

117 Aquí significa que cada correo electrónico puede pertenecer a varias carpetas temáticas al mismo tiempo.

Bibliografía

Para la realización de este proyecto ha sido necesario recurrir a un extenso material bibliográfico de diversa índole tales como libros, artículos en revistas científicas, artículos en conferencias, y páginas web.

AALTO UNIVERSITY (CIS), 2005. SOM Toolbox. [en línea]. [Consulta: 5 julio 2013]. Disponible en: <http://www.cis.hut.fi/projects/somtoolbox/>.

AIROLA, A., PAHIKKALA, T., WAEGEMAN, W., DE BAETS, B. y SALAKOSKI, T., 2009. A comparison of AUC estimators in small-sample studies. *3rd International workshop on Machine Learning in Systems Biology (MLSB 09)* [en línea]. S.l.: s.n., pp. 15–23. [Consulta: 8 octubre 2014]. Disponible en: <https://biblio.ugent.be/publication/860678/file/860679.pdf>.

ÁLVAREZ ROMERO, Y., DOMÍNGUEZ BRITO, A.C. y SUÁREZ ARAUJO, C.P., 2013. *Diversity in neural ensembles. Application in simultaneous resolution of fungicides* [en línea]. S.l.: Universidad de Las Palmas de Gran Canaria. [Consulta: 27 septiembre 2014]. Disponible en: http://acceda.ulpgc.es/bitstream/10553/11192/5/Tfm_Yeray_Alvarez_Romero.pdf.

ANDERSON, D., 2006. Statistical Spam Filtering. *EECS595* [en línea], vol. 4. [Consulta: 7 marzo 2013]. Disponible en: http://web.eecs.umich.edu/~rthomaso/courses/nlp/projects.06/David_Anderson.pdf.

ANDROUTSOPOULOS, I., KOUTSIAS, J., CHANDRINOS, K.V., PALIOURAS, G. y SPYROPOULOS, C.D., 2000. An evaluation of naive bayesian anti-spam filtering. *arXiv preprint cs/0006013* [en línea], [Consulta: 27 junio 2015]. Disponible en: <http://arxiv.org/abs/cs/0006013>.

ANDROUTSOPOULOS, I., KOUTSIAS, J., CHANDRINOS, K.V. y SPYROPOULOS, C.D., 2000. An experimental comparison of naive Bayesian and keyword-based anti-spam filtering with personal e-mail messages. *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval* [en línea]. S.l.: ACM, pp. 160–167. [Consulta: 27 junio 2015]. Disponible en: <http://dl.acm.org/citation.cfm?id=345569>.

ANDROUTSOPOULOS, I., PALIOURAS, G., KARKALETSIS, V., SAKKIS, G., SPYROPOULOS, C.D. y STAMATOPOULOS, P., 2000. Learning to filter spam e-mail: A comparison of a naive bayesian and a memory-based approach. *arXiv preprint cs/0009009* [en línea], [Consulta: 11 junio 2013]. Disponible en: <http://arxiv.org/abs/cs/0009009>.

ANDROUTSOPOULOS, I., PALIOURAS, G. y MICHELAKIS, E., 2004. *Learning to filter unsolicited commercial e-mail* [en línea]. S.l.: «DEMOKRITOS», National Center for Scientific Research. [Consulta: 27 junio 2015]. Disponible en: http://pages.cs.aueb.gr/nlp/pubs/TR2004_updated.pdf.

ANDROUTSOPOULOS, I., PALIOURAS, G., MICHELAKIS, E. y CSMINING.ORG, 2003. PU1 y PU123A. [en línea]. [Consulta: 9 abril 2013]. Disponible en: <http://www.csmining.org/index.php/pu1-and-pu123a-datasets.html>.

BASOGAIN OLABE, X., 1998. Redes Neuronales Artificiales y sus Aplicaciones. *Publicaciones de la Escuela de Ingenieros* [en línea], [Consulta: 5 marzo 2015]. Disponible en: http://datateca.unad.edu.co/contenidos/299018/11_-Redes_neuronales_artificiales_y_sus_aplicaciones.pdf.

BEKKERMAN, R., 2013. Ron Bekkerman: Email Classification on Enron Dataset. [en línea]. [Consulta: 9 abril 2013]. Disponible en: http://people.cs.umass.edu/~ronb/enron_dataset.html.

BEKKERMAN, R., MCCALLUM, A. y HUANG, G., 2004. Automatic categorization of email into folders: Benchmark experiments on Enron and SRI corpora. *Center for Intelligent Information Retrieval (CIIR) - University of Massachusetts* [en línea], vol. Technical Report IR-418 2004.

[Consulta: 1 julio 2015]. Disponible en: http://scholarworks.umass.edu/cs_faculty_pubs/218/?utm_source=scholarworks.umass.edu%2Fcs_faculty_pubs%2F218&utm_medium=PDF&utm_campaign=PDFCoverPages.

BLANCO, Á., RICKET, A. y MARTÍN-MERINO, M., 2007. Combining SVM classifiers for email anti-spam filtering. *Computational and Ambient Intelligence*, pp. 903–910 (10).

BOROVICKA, T., JIRINA JR., M., KORDIK, P. y JIRINA, M., 2012. Selecting Representative Data Sets. *Advances in Data Mining Knowledge Discovery and Applications* [en línea]. S.l.: Associate Prof. Adem Karahoca, ISBN 978-953-51-0748-4. Disponible en: <http://www.intechopen.com/books/advances-in-data-mining-knowledge-discovery-and-applications/selecting-representative-data-sets>.

BRADLEY, A.P., 1997. The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition*, vol. 30, no. 7, pp. 1145-1159.

BRIGHAM YOUNG UNIVERSITY, 1994. British National Corpus (BYU-BNC). [en línea]. [Consulta: 18 diciembre 2013]. Disponible en: <http://corpus.byu.edu/bnc/>.

BROWN, C.D. y DAVIS, H.T., 2006. Receiver operating characteristics curves and related decision measures: A tutorial. *Chemometrics and Intelligent Laboratory Systems*, vol. 80, no. 1, pp. 24-38. ISSN 01697439. DOI 10.1016/j.chemolab.2005.05.004.

BULLINARIA, J.A., 2012a. Self Organizing Maps: Algorithms and Applications. [en línea]. S.l. [Consulta: 25 mayo 2013]. Disponible en: <http://www.cs.bham.ac.uk/~jxb/INC/l17h.pdf>.

BULLINARIA, J.A., 2012b. Self Organizing Maps: Fundamentals. [en línea]. S.l. [Consulta: 25 mayo 2013]. Disponible en: <http://www.cs.bham.ac.uk/~jxb/INC/l16.pdf>.

CABESTANY, J., IENNE, P., MORENO, J.M. y MADRENAS, J., 1996. Is There a future for ANN Hardware. *Workshop on Mixed Design of Integrated Circuits and Systems, Poland, Lodz* [en línea]. S.l.: Citeseer, [Consulta: 8 marzo 2015]. Disponible en: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.7.8418&rep=rep1&type=pdf>.

CABRERA, Y. y ACOSTA, O., 2011. spam. . Politechnika Wroclawska (Poland): INE4242.

CHAPELLE, O., SCHÖLKOPF, B. y ZIEN, A., 2006. *Semi-supervised learning* [en línea]. S.l.: MIT press Cambridge, MA: [Consulta: 11 diciembre 2012]. Disponible en: [http://fs1.bib.tiera.ru/content/dvd38/Chapelle%20O.%20\(Ed\),%20Scholkopf%20B.%20\(Ed\),%20Zien%20A.%20\(Ed\)%20-%20Semi-Supervised%20Learning\(2006\)\(508\).pdf](http://fs1.bib.tiera.ru/content/dvd38/Chapelle%20O.%20(Ed),%20Scholkopf%20B.%20(Ed),%20Zien%20A.%20(Ed)%20-%20Semi-Supervised%20Learning(2006)(508).pdf).

CHHABRA, P., WADHVANI, R. y SHUKLA, S., 2010. Spam Filtering using Support Vector Machine. , pp. 6.

CHHABRA, S., 2005. *Fighting spam, phishing and email fraud* [en línea]. S.l.: University of California Riverside. [Consulta: 3 febrero 2015]. Disponible en: <http://www.cs.ucr.edu/~schhabra/thesis.pdf>.

CHUAN, Z., XIANLIANG, L., MENGSHU, H. y XU, Z., 2005. A LVQ-based neural network anti-spam email approach. *ACM SIGOPS Operating Systems Review*, vol. 39, no. 1, pp. 34–39 (6).

CLARK, J., KOPRINSKA, I. y POON, J., 2003. A neural network based approach to automated e-

mail classification. *Web Intelligence, 2003. WI 2003. Proceedings. IEEE/WIC International Conference on* [en línea]. S.l.: s.n., pp. 702–705 (4). [Consulta: 27 noviembre 2012]. Disponible en: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1241300.

COHEN, W.W., 2013. Enron Email Dataset. [en línea]. [Consulta: 9 abril 2013]. Disponible en: <http://www.cs.cmu.edu/~enron/>.

CORMACK, G.V. y MOJDEH, M., 2009. Machine learning for information retrieval: TREC 2009 web, relevance feedback and legal tracks. *The eighteenth Text REtrieval Conference proceedings (TREC 2009), Gaithersburg, MD* [en línea]. S.l.: s.n., [Consulta: 6 diciembre 2012]. Disponible en: <http://trec.nist.gov/pubs/trec18/papers/uwaterloo-cormack.WEB.RF.LEGAL.pdf>.

CORTES GENERALES DE ESPAÑA, 1999. Ley Orgánica 15/1999, de 13 de diciembre, de Protección de Datos de Carácter Personal. *BOE de* [en línea], vol. 14. [Consulta: 19 septiembre 2014]. Disponible en: <http://www.ugtextremadura.org/userfiles/ugt73.pdf>.

CORTES GENERALES DE ESPAÑA, 2002. Ley 34/2002, de 11 de julio, de Servicios de la Sociedad de la Información y de Comercio Electrónico. *Nº166* [en línea], vol. 12. [Consulta: 19 septiembre 2014]. Disponible en: <http://www.wipo.int/edocs/lexdocs/laws/es/es/es159es.pdf>.

DRUCKER, H., WU, D. y VAPNIK, V.N., 1999. Support vector machines for spam categorization. *Neural Networks, IEEE Transactions on*, vol. 10, no. 5, pp. 1048–1054 (7).

DUDA, R.O., HART, P.E. y STORK, D.G., 2001. *Pattern classification* [en línea]. 2. S.l.: Wiley. [Consulta: 26 septiembre 2014]. Disponible en: http://books.google.es/books/about/Pattern_classification.html?id=YoxQAAAAMAAJ.

ELUYODE, O.S. y AKOMOLAFE, D.T., 2013. Comparative study of biological and artificial neural networks. [en línea], [Consulta: 10 septiembre 2014]. Disponible en: <http://scholarsresearchlibrary.com/EJAESR-vol2-iss1/EJAESR-2013-2-1-36-46.pdf>.

ERICKSON, D., CASADO, M. y MCKEOWN, N., 2008. The Effectiveness of Whitelisting: a User-Study. *Proc. of Conference on Email and Anti-Spam* [en línea]. S.l.: s.n., pp. 10. [Consulta: 8 diciembre 2012]. Disponible en: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.151.9251&rep=rep1&type=pdf>.

ERWIN, E., OBERMAYER, K. y SCHULTEN, K., 1991. Convergence Properties of Self-Organizing Maps. *Artificial Neural Networks* [en línea], [Consulta: 12 junio 2015]. Disponible en: <http://ottawa.ks.uiuc.edu/Publications/Papers/PDF/ERWI91/ERWI91.pdf>.

ERWIN, E., OBERMAYER, K. y SCHULTEN, K., 1992. Self-organizing maps: ordering, convergence properties and energy functions. *Biological cybernetics*, vol. 67, no. 1, pp. 47–55.

FAWCETT, T., 2004. ROC graphs: Notes and practical considerations for researchers. *Machine learning*, vol. 31, pp. 1–38.

FAWCETT, T., 2006. An introduction to ROC analysis. *Pattern recognition letters*, vol. 27, no. 8, pp. 861–874.

FORCADA, V.R., 2003. *Clasificación supervisada basada en redes bayesianas, aplicación en biología computacional* [en línea]. S.l.: Universidad Politécnica de Madrid. [Consulta: 20 febrero 2015]. Disponible en: <http://dialnet.unirioja.es/servlet/tesis?codigo=3037>.

- FORT, J.-C., LETREMY, P. y COTTRELL, M., 2002. Advantages and drawbacks of the Batch Kohonen algorithm. *ESANN* [en línea]. S.l.: s.n., pp. 223–230. [Consulta: 28 septiembre 2014]. Disponible en: <http://samos.univ-paris1.fr/archives/ftp/preprints/samos158.pdf>.
- FRESCHI, V., SERAGHITI, A. y BOGLIOLO, A., 2006. Filtering obfuscated email spam by means of phonetic string matching. *Advances in Information Retrieval* [en línea]. S.l.: Springer, pp. 505–509. [Consulta: 29 marzo 2014]. Disponible en: http://link.springer.com/chapter/10.1007/11735106_52.
- FREUND, Y. y SCHAPIRE, R.E., 1999. Large margin classification using the perceptron algorithm. *Machine learning*, vol. 37, no. 3, pp. 277–296 (19).
- FUMERA, G., PILLAI, I. y ROLI, F., 2006. Spam filtering based on the analysis of text information embedded into images. En: R. LIPPMANN (ed.), *The Journal of Machine Learning Research*, vol. 7, pp. 2699–2720.
- GABRIELSSON, S. y GABRIELSSON, S., 2006. *The Use of self-organizing maps in recommender systems. A survey of the Recommender Systems field and a presentation of a State of the Art Highly Interactive Visual Movie Recommender System* [en línea]. Master's Thesis. S.l.: Department of Information Technology at the Division of Computer Systems, Uppsala University. [Consulta: 27 julio 2014]. Disponible en: <http://rslab.movsom.com/paper/somrs/somrs.pdf>.
- GAMMERMAN, A., VOVK, V. y VAPNIK, V.N., 2013. *Learning by Transduction* [en línea]. 2013. S.l.: s.n. [Consulta: 29 enero 2015]. Disponible en: <http://www1.cs.columbia.edu/~dpolewis/candidacy/gammerman98learning.pdf>.
- GAO, Y., YAN, M. y CHOUDHARY, A., 2009. Semi Supervised Image Spam Hunter: A Regularized Discriminant EM Approach. [en línea]. S.l.: s.n., pp. 152–164 (13). [Consulta: 29 noviembre 2012]. Disponible en: http://users.eecs.northwestern.edu/~mya671/mypapers/ADMA09_Gao_Yang_Choudhary.pdf.
- GARCÍA ABAD, J., 2012. *Calibración Local de Predicciones Numéricas de Viento con Técnicas Estadísticas no Lineales (Downscaling Estadístico)* [en línea]. Santander: Universidad de Cantabria. [Consulta: 5 junio 2015]. Disponible en: <http://repositorio.unican.es/xmlui/bitstream/handle/10902/1006/Jesus%20Garcia%20Abad.pdf?sequence=1&isAllowed=y>.
- GARCIA, F., HOEPMAN, J.H. y VAN NIEUWENHUIZEN, J., 2004. Spam filter analysis. *Security and Protection in Information Processing Systems*, pp. 395–410 (17).
- GOLBECK, J. y HENDLER, J., 2004. Reputation network analysis for email filtering. *Proceedings of the First Conference on Email and Anti-Spam* [en línea]. S.l.: s.n., pp. 54–58. [Consulta: 6 diciembre 2012]. Disponible en: <ftp://ftp.research.microsoft.com/Users/joshuago/conference/papers-2004/177.pdf>.
- GRAHAM-CUMMING, J., 2006. SpamOrHam. , pp. 22-24 (2).
- GUENTER, B., 1998. SPAM Archive. [en línea]. [Consulta: 2 julio 2015]. Disponible en: <http://untroubled.org/spam/>.
- GUZELLA, T.S. y CAMINHAS, W.M., 2009. A review of machine learning approaches to Spam filtering. *Expert Systems with Applications*, vol. 36, no. 7, pp. 10206-10222. ISSN 09574174. DOI

10.1016/j.eswa.2009.02.037.

HAYKIN, S.S., 1999. *Neural Networks. A Comprehensive Foundation* [en línea]. 2nd. Ontario, Canada: Prentice-Hall International. [Consulta: 27 septiembre 2014]. Disponible en: <http://www.mif.vu.lt/~valdas/DNT/Literatura/Haykin99/Neural%20Networks-A%20Comprehensive%20Foundation,%202nd%20Edition%20by%20Simon%20S.%20Haykin.pdf>.

HAYKIN, S.S., 2009. *Neural Networks and Learning Machines*. S.l.: Prentice Hall. ISBN 978-0-13-147139-9.

HEBB, D.O., 1949. *The organization of behavior: A neuropsychological theory* [en línea]. S.l.: Psychology Press. [Consulta: 10 septiembre 2014]. Disponible en: [http://books.google.com/books?hl=en&lr=&id=uyV5AgAAQBAJ&oi=fnd&pg=PP1&dq=%22to+bridge+the+gap+between+neurophysiology+and%22+%22frequently+have+direct+access+to+data+of+first%22+%22human+adult\)+psychology,+and+Moss+\(1942\)+or+Maier%22+%22winter.+Part+or+all+of+the+manuscript+was+read+by%22+&ots=mImSArJTt&sig=HwZvBVhEWJGyHFjibReIbrHXNFc](http://books.google.com/books?hl=en&lr=&id=uyV5AgAAQBAJ&oi=fnd&pg=PP1&dq=%22to+bridge+the+gap+between+neurophysiology+and%22+%22frequently+have+direct+access+to+data+of+first%22+%22human+adult)+psychology,+and+Moss+(1942)+or+Maier%22+%22winter.+Part+or+all+of+the+manuscript+was+read+by%22+&ots=mImSArJTt&sig=HwZvBVhEWJGyHFjibReIbrHXNFc).

HILL, S., 1999. samhill.co.uk | Kohonen Neural Networks. [en línea]. [Consulta: 9 diciembre 2012]. Disponible en: <http://www.samhill.co.uk/kohonen/>.

HOPKINS, M., REEBER, E., FORMAN, G. y SUERMONDT, J., 1999. UCI Machine Learning Repository: Spambase Data Set. [en línea]. [Consulta: 9 abril 2013]. Disponible en: <http://archive.ics.uci.edu/ml/datasets/Spambase>.

HOVOLD, J., 2005. Naive Bayes Spam Filtering Using Word-Position-Based Attributes. *CEAS* [en línea]. S.l.: s.n., [Consulta: 6 agosto 2013]. Disponible en: <http://fileadmin.cs.lth.se/ai/psfiles/ceas-05b.pdf>.

INDRAYAN, A., 2012. *ROC Curve - Medical Biostatistics* [en línea]. 3ª. S.l.: Chapman & Hall. [Consulta: 17 febrero 2015]. Disponible en: <http://www.medicalbiostatistics.com/roccurve.pdf>.

JAIN, A.K., MAO, J. y MOHIUDDIN, K.M., 1996. Artificial Neural Networks: A Tutorial. *IEEE Computational Science & Engineering*, vol. 29, pp. 31-44.

JAMES BRUCE, 2012. Grey Mail: The New Email Nuisance To Hit Your Inbox. [en línea]. [Consulta: 4 marzo 2013]. Disponible en: <http://www.makeuseof.com/tag/grey-mail-new-email-nuisance-inbox/>.

JOACHIMS, T., 1999. Transductive Inference for Text Classification using Support Vector Machines. [en línea]. S.l.: s.n., pp. 10. [Consulta: 11 diciembre 2012]. Disponible en: <http://www1.cs.columbia.edu/~dplewis/candidacy/joachims99transductive.pdf>.

KARAHOCA, A. (ed.), 2012. *Advances in Data Mining Knowledge Discovery and Applications* [en línea]. S.l.: InTech. [Consulta: 23 septiembre 2014]. ISBN 978-953-51-0748-4. Disponible en: <http://www.intechopen.com/books/advances-in-data-mining-knowledge-discovery-and-applications>.

KASKI, S. y LAGUS, K., 1996. Comparing Self-Organizing Maps. *ICANN96* [en línea]. Berlin: Springer, pp. 809-814. [Consulta: 21 julio 2014]. Disponible en: <http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=44E93668DE4AC3CC4EA5A828F702CABF?doi=10.1.1.47.8238&rep=rep1&type=pdf>.

- KEILA, P.S. y SKILLICORN, D.B., 2005a. Detecting unusual and deceptive communication in email. *Centers for Advanced Studies Conference* [en línea]. S.l.: s.n., pp. 17–20. [Consulta: 1 julio 2015]. Disponible en: <http://research.cs.queensu.ca/TechReports/Reports/2005-498.pdf>.
- KEILA, P.S. y SKILLICORN, D.B., 2005b. Structure in the Enron email dataset. *Computational & Mathematical Organization Theory*, vol. 11, no. 3, pp. 183–199.
- KELLER, F., 2003. Connectionist and Statistical Language Processing. *Universität des Saarlandes*, [Online]. Available: <http://www.coli.unisaarland.de/crocker/courses/learning/lecture10.pdf> [Accessed: July 2012] [en línea], [Consulta: 27 mayo 2014]. Disponible en: <http://www.coli.uni-saarland.de/~crocker/courses/learning/lecture11.pdf>.
- KHORSI, A., 2007. An overview of content-based spam filtering techniques. *INFORMATICA-LJUBLJANA*-, vol. 31, no. 3, pp. 269 (10).
- KIVILUOTO, K., 1996. Topology Preservation in Self-Organizing Maps. *International Conference on Neural Networks* [en línea]. <http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=548907>: s.n., [Consulta: 7 agosto 2014]. Disponible en: <http://cis.legacy.ics.tkk.fi/kkluoto/publications/icnn96.pdf>.
- KLENSIN, J., 2008. RFC 5321 - Simple Mail Transfer Protocol. [en línea]. [Consulta: 17 septiembre 2014]. Disponible en: <http://tools.ietf.org/html/rfc5321>.
- KLIMT, B. y YANG, Y., 2004. The Enron Corpus: A new dataset for email classification research. *Machine learning: ECML 2004* [en línea]. S.l.: Springer, pp. 217–226. [Consulta: 26 septiembre 2014]. Disponible en: http://link.springer.com/chapter/10.1007/978-3-540-30115-8_22.
- KOHONEN, T., 1988. An Introduction to Neural Computing. *Neural Networks*1, pp. 3-16.
- KOHONEN, T., 2001. *Self-Organizing Maps - Teuvo Kohonen - Google Libros* [en línea]. S.l.: Springer. [Consulta: 21 julio 2014]. Disponible en: <http://books.google.es/books?id=e4igHzyfO78C&printsec=frontcover&hl=es#v=onepage&q&f=false>.
- KOHONEN, T., 2013. Essentials of the self-organizing map. *Neural Networks*, vol. 37, pp. 52-65. ISSN 08936080. DOI 10.1016/j.neunet.2012.09.018.
- KOHONEN, T., HYNINEN, J., KANGAS, J. y LAAKSONEN, J., 1996. SOM_PAK: The Self-Organizing Map Program Package. *Report A31, Helsinki University of Technology, Laboratory of Computer and Information Science* [en línea], [Consulta: 26 diciembre 2014]. Disponible en: http://hackbbs.org/article/reds/form_de_base/SBI/labo5/som_pak.pdf.
- KOHONEN, T., NIEMINEN, I.T. y HONKELA, T., 2009. On the quantization error in SOM vs. VQ: A critical and systematic study. *Advances in Self-Organizing Maps* [en línea]. S.l.: Springer, pp. 133–144. [Consulta: 14 agosto 2014]. Disponible en: http://link.springer.com/chapter/10.1007/978-3-642-02397-2_16.
- KOLCZ, A., CHOWDHURY, A. y ALSPECTOR, J., 2004. The impact of feature selection on signature-driven spam detection. *Proceedings of the 1st Conference on Email and Anti-Spam (CEAS-2004)* [en línea]. S.l.: s.n., [Consulta: 6 diciembre 2012]. Disponible en: <http://ir.cs.georgetown.edu/~abdur/publications/147.pdf>.
- KOTSIANTIS, S.B., KANELLOPOULOS, D. y PINTELAS, P.E., 2006. Data preprocessing for

supervised learning. *International Journal of Computer Science*, vol. 1, no. 2, pp. 111–117.

KUFANDIRIMBWA, O. y GOTOR, R., 2012. Spam Detection Using Artificial Neural Networks (Perceptron Learning Rule). *Online Journal of Physical and Environmental Science Research*, vol. 1, no. 2, pp. 22–29.

LANDAUER, T. y DUMAIS, S., 2008. Latent semantic analysis. *Scholarpedia*, vol. 3, no. 11, pp. 4356. ISSN 1941-6016. DOI 10.4249/scholarpedia.4356.

LEOW, P., 2014. Self-organizing Map Demystified - CodeProject. [en línea]. [Consulta: 7 agosto 2014]. Disponible en: <http://www.codeproject.com/Articles/789733/Self-organizing-Map-Demystified>.

LERTNATTEE, V. y THEERAMUNKONG, T., 2004. Analysis of inverse class frequency in centroid-based text classification. *Communications and Information Technology, 2004. ISCIT 2004. IEEE International Symposium* [en línea]. S.l.: s.n., [Consulta: 21 febrero 2014]. Disponible en: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1413903.

LIEB, R., 2002. Make Spammers Pay Before You Do | ClickZ. [en línea]. [Consulta: 11 octubre 2014]. Disponible en: <http://www.clickz.com/clickz/column/2138759/make-spammers-pay-before-you-do>.

LINGUIST, 2013. LINGUIST List | Home. [en línea]. [Consulta: 9 abril 2013]. Disponible en: <http://linguistlist.org/>.

LIU, C. y STAMM, S., 2007. Fighting unicode-obfuscated spam. *Proceedings of the anti-phishing working groups 2nd annual eCrime researchers summit* [en línea]. S.l.: ACM, pp. 45–59. [Consulta: 29 marzo 2014]. Disponible en: <http://dl.acm.org/citation.cfm?id=1299020>.

LOBO, J.M., JIMÉNEZ-VALVERDE, A. y REAL, R., 2008. AUC: a misleading measure of the performance of predictive distribution models. *Global Ecology and Biogeography*, vol. 17, no. 2, pp. 145–151. ISSN 1466-822X, 1466-8238. DOI 10.1111/j.1466-8238.2007.00358.x.

LÓPEZ, V., FERNÁNDEZ, A., MORENO-TORRES, J.G. y HERRERA, F., 2012. Analysis of preprocessing vs. cost-sensitive learning for imbalanced classification. Open problems on intrinsic data characteristics. *Expert Systems with Applications*, vol. 39, no. 7, pp. 6585–6608. ISSN 09574174. DOI 10.1016/j.eswa.2011.12.043.

LOWD, D. y MEEK, C., 2005. Good Word Attacks on Statistical Spam Filters. *CEAS* [en línea]. S.l.: s.n., [Consulta: 28 enero 2015]. Disponible en: http://pdf.aminer.org/000/085/108/good_word_attacks_on_statistical_spam_filters.pdf.

LUO, X. y ZINCIR-HEYWOOD, N., 2005. Comparison of a SOM based sequence analysis system and naive Bayesian classifier for spam filtering. *Neural Networks, 2005. IJCNN'05. Proceedings. 2005 IEEE International Joint Conference on* [en línea]. S.l.: s.n., pp. 2571–2576 (6). [Consulta: 9 diciembre 2012]. Disponible en: <http://web.cs.dal.ca/~luo/ijcnn05-xn-final.pdf>.

LUTUS, P., 2006. The Anti-Spam Home Page. [en línea]. [Consulta: 27 enero 2015]. Disponible en: <http://www.arachnoid.com/lutusp/antispam.html>.

MALATHI, R., 2011. Email Spam Filter using Supervised Learning with Bayesian Neural Network. , pp. 12.

- MARTINETZ, T. y SCHULTEN, K., 1991. A «neural gas» network learns topologies. *Artificial Neural Networks*, vol. Elsevier, pp. 397-402.
- MATHWORKS, 2014. *Parallel Computing Toolbox for Matlab R2014a - User's Guide* [en línea]. 2014. S.l.: s.n. [Consulta: 11 julio 2014]. Disponible en: http://www.mathworks.com/help/pdf_doc/distcomp/distcomp.pdf.
- MCAFEE y ICF INTERNATIONAL, 2009. The Carbon Footprint of Email Spam Report. [en línea]. S.l.: [Consulta: 4 noviembre 2013]. Disponible en: http://img.en25.com/Web/McAfee/CarbonFootprint_web_final2.pdf.
- MCCARTOR, H., 1991. Back Propagation Implementation on the Adaptive Solutions CNAPS Neurocomputer Chip. *Advances in Neural Information Processing Systems* [en línea]. S.l.: s.n., pp. 1028–1031. [Consulta: 8 marzo 2015]. Disponible en: <http://papers.nips.cc/paper/383-back-propagation-implementation>.
- METSIS, V., ANDROUTSOPOULOS, I. y PALIOURAS, G., 2006a. Spam filtering with naive bayes-which naive bayes? *CEAS* [en línea]. S.l.: s.n., pp. 27–28. [Consulta: 2 agosto 2013]. Disponible en: http://classes.soe.ucsc.edu/cmeps242/Fall09/lect/12/CEAS2006_corrected-naiveBayesSpam.pdf.
- METSIS, V., ANDROUTSOPOULOS, I. y PALIOURAS, G., 2006b. Spam Filtering with Naive Bayes – Which Naive Bayes? [en línea]. S.l. [Consulta: 2 agosto 2013]. Disponible en: http://www.aueb.gr/users/ion/docs/ceas_2006_slides.pdf.
- METZ, C.E., 1978. Basic principles of ROC analysis. *Seminars in nuclear medicine* [en línea]. S.l.: Elsevier, pp. 283–298. [Consulta: 5 febrero 2015]. Disponible en: <http://www.sciencedirect.com/science/article/pii/S0001299878800142>.
- MIKKULAINEN, R., 2010. Topology of a Neural Network. En: C. SAMMUT y G.I. WEBB (eds.), *Encyclopedia of Machine Learning* [en línea]. S.l.: Springer, pp. 988-989. [Consulta: 3 junio 2015]. ISBN 978-0-387-30164-8. Disponible en: http://link.springer.com/referenceworkentry/10.1007%2F978-0-387-30164-8_837.
- MIKKELSEN, J.A., 2006. *Self-organizing Neural Networks* [en línea]. Denmark: University of Southern Denmark. Disponible en: grydeske.net/download/index/3?errorAction=upload&errorController=admin.
- MINKOV, E., WANG, R.C. y COHEN, W.W., 2005. Extracting personal names from email: Applying named entity recognition to informal text. *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing* [en línea]. S.l.: Association for Computational Linguistics, pp. 443–450. [Consulta: 1 julio 2015]. Disponible en: <http://dl.acm.org/citation.cfm?id=1220631>.
- MOJDEH, M., 2012. *Personal Email Spam Filtering with Minimal User Interaction* [en línea]. S.l.: s.n. [Consulta: 29 noviembre 2012]. Disponible en: http://uwspace.uwaterloo.ca/bitstream/10012/6675/1/Mojdeh_Mona.pdf.
- MOJDEH, M. y CORMACK, G.V., 2008. Semi-supervised spam filtering: does it work? (abstract). *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval* [en línea]. S.l.: s.n., pp. 745–746 (2). [Consulta: 27 noviembre 2012]. Disponible en: <http://dl.acm.org/citation.cfm?id=1390482>.

NARISAWA, K., BANNAI, H., HATANO, K. y TAKEDA, M., 2007. Unsupervised spam detection based on string alienness measures. *Discovery Science* [en línea]. S.l.: s.n., pp. 161–172 (12). [Consulta: 27 noviembre 2012]. Disponible en: <http://www.springerlink.com/index/a85hul2663443776.pdf>.

NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY, 2013. Text REtrieval Conference (TREC) Home Page. [en línea]. [Consulta: 9 abril 2013]. Disponible en: <http://trec.nist.gov/>.

NAVAN, 2014. Understanding ROC curves. [en línea]. [Consulta: 17 abril 2015]. Disponible en: <http://www.navan.name/roc/>.

OBERG, N., RUDELLE, B., GARCÍA, M.H. y KUMAR, P., 2008. MATLAB Parallel Computing Toolbox Benchmark for an Embarrassingly Parallel Application. [en línea]. Urbana, IL: Department of Civil and Environmental Engineering - University of Illinois. [Consulta: 26 junio 2015]. Disponible en: http://vtchl.uiuc.edu/sites/default/files/MATLAB_Report.pdf.

PADHYE, A. y PEDERSEN, T., 2006a. *Comparing Supervised and Unsupervised Classification of Messages in the Enron Email Corpus* [en línea]. S.l.: University of Minnesota. [Consulta: 11 marzo 2013]. Disponible en: <http://d.umn.edu/~tpederse/Pubs/apurva-thesis.pdf>.

PADHYE, A. y PEDERSEN, T., 2006b. Ted Pedersen - Enron Email Corpus by Topic. [en línea]. [Consulta: 9 abril 2013]. Disponible en: <http://www.d.umn.edu/~tpederse/enron.html>.

PALIN, S. y SUNLIGHT FOUNDATION, 2009. Sarah's Inbox | A Project of the Sunlight Foundation. [en línea]. [Consulta: 9 abril 2013]. Disponible en: <http://sarahsinbox.com/>.

PARLAMENTO EUROPEO y CONSEJO, 2002. DIRECTIVA 2002/58/CE DEL PARLAMENTO EUROPEO Y DEL CONSEJO, de 12 de julio de 2002, relativa al tratamiento de los datos personales y a la protección de la intimidad en el sector de las comunicaciones electrónicas (Directiva sobre la privacidad y las comunicaciones electrónicas). *Diario Oficial de las Comunidades Europeas* [en línea], [Consulta: 27 enero 2015]. Disponible en: <http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=OJ:L:2002:201:0037:0047:es:PDF>.

PFAHRINGER, B., 2006. A semi-supervised spam mail detector. , pp. 5.

PITSILLIDIS, A., LEVCHENKO, K., KREIBICH, C., KANICH, C., VOELKER, G.M., PAXSON, V., WEAVER, N. y SAVAGE, S., 2010. Botnet judo: Fighting spam with itself. *Symposium on Network and Distributed System Security (NDSS)* [en línea]. S.l.: s.n., pp. 19. [Consulta: 27 noviembre 2012]. Disponible en: <http://www.dcg.ethz.ch/lectures/seminar/paper/Samuel/22.pdf>.

PÖLZLBAUER, G., 2004. *Survey and comparison of quality measures for self-organizing maps* [en línea]. S.l.: na. [Consulta: 21 julio 2014]. Disponible en: <http://www.ifs.tuwien.ac.at/~poelzlbauer/publications/Poe04WDA.pdf>.

PORTER, M., 1980. An algorithm for suffix stripping. *Program*, vol. 14, no. 3, pp. 130-137.

POSTINI, 2004. *The shifting tactics of spammers: What you need to know about new email threats* [en línea]. 2004. S.l.: s.n. [Consulta: 28 enero 2015]. Disponible en: http://secureemailplus.com/wp/WP10-01-0406_Postini_Connections.pdf.

POWERS, D.M., 2011. Evaluation: from precision, recall and F-measure to ROC, informedness,

markedness and correlation. [en línea], [Consulta: 18 febrero 2015]. Disponible en: <http://dspace2.flinders.edu.au/xmlui/handle/2328/27165>.

P. RESNICK, E., 2008. RFC 5322 - Internet Message Format. [en línea]. [Consulta: 19 diciembre 2014]. Disponible en: <http://tools.ietf.org/html/rfc5322>.

PROVOST, F., FAWCETT, T. y KOHAVI, R., 1997. The Case Against Accuracy Estimation for Comparing Induction Algorithms. In *Proceedings of the Fifteenth International Conference on Machine Learning* [en línea]. S.l.: s.n., [Consulta: 23 febrero 2015]. Disponible en: <http://eecs.wsu.edu/~holder/courses/cse6363/spr04/pubs/Provost98.pdf>.

QIAN, F., PATHAK, A., HU, Y.C., MAO, Z.M. y XIE, Y., 2010. A Case for Unsupervised-Learning-based Spam Filtering. [en línea]. S.l.: s.n., pp. 17. [Consulta: 27 noviembre 2012]. Disponible en: <https://www.eecs.umich.edu/techreports/cse/2010/CSE-TR-561-10.pdf>.

RAE, 2005. arte --> estado del arte - Diccionario panhispánico de dudas | Real Academia Española. [en línea]. [Consulta: 21 diciembre 2014]. Disponible en: <http://lema.rae.es/dpd/?key=arte>.

RAMACHANDRAN, A. y FEAMSTER, N., 2006. Understanding the network-level behavior of spammers. *ACM SIGCOMM Computer Communication Review* [en línea]. S.l.: s.n., pp. 291–302. [Consulta: 6 diciembre 2012]. Disponible en: <http://dl.acm.org/citation.cfm?id=1159947>.

RAO, J.M. y REILEY, D.H., 2012. The Economics of Spam. *Journal of Economic Perspectives*, vol. 26, no. 3, pp. 87-110. ISSN 0895-3309. DOI 10.1257/jep.26.3.87.

ROCA, M.D., DOMÍNGUEZ, J.D.G., FIGUEROA, Z.J.H., AGUIAR, J.R.P., PINO, J.C.R. del y RODRÍGUEZ, G.R., 2005. *Fundamentos de estructuras de datos. Soluciones en Ada, Java y C++* [en línea]. Madrid: Thomson Paraninfo. Disponible en: <http://www.paraninfo.es/catalogo/9788497323581/fundamentos-de-estructuras-de-datos--soluciones-en-ada--java-y-c-->.

ROJAS, R., 1999. *Kohonen Networks in Neural Networks - A Systematic Introduction* [en línea]. Berlín: Springer-Verlag. [Consulta: 26 mayo 2013]. Disponible en: <http://page.mi.fu-berlin.de/rojas/neural/chapter/K15.pdf>.

RONG JIN, JOYCE Y. CHAI y LUO SI, 2005. Learn to Weight Terms in Information Retrieval Using Category Information. *Proceedings of the 22 nd International Conference on Machine Learning* [en línea]. Bonn, Germany: s.n., [Consulta: 20 febrero 2014]. Disponible en: http://machinelearning.org/proceedings/icml2005/papers/045_LearnToWeight_JinEtAl.pdf.

RUMELHART, D.E., HINTON, G.E. y MCCLELLAND, J.L., 1986. *A General Framework for Parallel Distributed Processing* [en línea]. Cambridge, MA, USA: MIT Press. [Consulta: 6 marzo 2015]. ISBN 0-262-68053-X. Disponible en: <http://www.cs.toronto.edu/~fritz/absps/pdp2.pdf>.

SAHAMI, M., DUMAIS, S., HECKERMAN, D. y HORVITZ, E., 1998. A Bayesian approach to filtering junk e-mail. *Learning for Text Categorization: Papers from the 1998 workshop* [en línea]. S.l.: s.n., pp. 98–105 (8). [Consulta: 27 noviembre 2012]. Disponible en: <http://www.searchforum.org.cn/dataflowgroup/Reading/2004Paper/SpamPaper/ContentandStatistic al/A%20Bayesian%20Approach%20to%20Filtering%20Junk%20E-Mail.pdf>.

SANTIAGO RAMÓN Y CAJAL, 1917. *Recuerdos de mi vida. Láminas ilustrativas*. [en línea]. 1917. S.l.: Centro Virtual Cervantes. [Consulta: 4 marzo 2015]. Disponible en:

http://cvc.cervantes.es/ciencia/cajal/cajal_recuernos/recuernos/laminas.htm.

SANTOS, I., SANZ, B., LAORDEN, C., BREZO, F. y BRINGAS, P.G., 2011. Computational Intelligence in Security for Information Systems: 4th ... - Google Libros. [en línea]. [Consulta: 11 diciembre 2012]. Disponible en: http://books.google.es/books?id=IFwvAqxXf3wC&pg=PA53&lpg=PA53&dq=llgc+algorithm&source=bl&ots=jSwG9XaV_h&sig=u6P3Z3bEKBF-nwk_bx-swIrExZs&hl=es&sa=X&ei=24DHUNKiNoGMygH08ICwDA&ved=0CG0Q6AEwCA#v=onepage&q=llgc%20algorithm&f=false.

SASSI, R.J., 2011. An hybrid architecture for clusters analysis: rough sets theory and Self-Organizing Map Artificial Neural Network. [en línea], [Consulta: 7 agosto 2014]. Disponible en: <http://www.scielo.br/pdf/pope/v32n1/aop0512.pdf>.

SAYAD, S., 2010. Self-Organizing Maps (SOM). [en línea]. S.I. [Consulta: 25 mayo 2013]. Disponible en: <http://chem-eng.utoronto.ca/~datamining/Presentations/SOM.pdf>.

SCHATZ, C.J., 1992. The Developing Brain. *Scientific American*, vol. 267, no. 3, pp. 60-67. ISSN 0036-8733.

SCHNEIDER, K.-M., 2003. A comparison of event models for Naive Bayes anti-spam e-mail filtering. *Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics-Volume 1* [en línea]. S.I.: s.n., pp. 307–314. [Consulta: 11 junio 2013]. Disponible en: <http://dl.acm.org/citation.cfm?id=1067848>.

SCULLEY, D., WACHMAN, G. y BRODLEY, C.E., 2006. Spam Filtering Using Inexact String Matching in Explicit Feature Space with On-Line Linear Classifiers. *TREC* [en línea]. S.I.: s.n., [Consulta: 28 enero 2015]. Disponible en: <http://www.eecs.tufts.edu/~dsculley/papers/trec.2006.spam.pdf>.

SHAFRANOVICH, Y., 2005. RFC 4180 - Common Format and MIME Type for Comma-Separated Values (CSV) Files. [en línea]. [Consulta: 20 mayo 2013]. Disponible en: <http://tools.ietf.org/html/rfc4180>.

SHUNLI, Z. y QINGSHUANG, Y., 2010. Personal Spam Filter by Semi-supervised Learning. , pp. 4.

SINHA, S., SINGH, T.N., SINGH, V.K. y VERMA, A.K., 2010. Epoch determination for neural network by self-organized map (SOM). *Computational Geosciences*, vol. 14, no. 1, pp. 199-206. ISSN 1420-0597, 1573-1499. DOI 10.1007/s10596-009-9143-0.

SIVANANDAM, S.N. y DEEPA, S.N., 2006. *Introduction to Neural Networks Using Matlab 6.0*. S.I.: Tata McGraw-Hill Education. ISBN 978-0-07-059112-7.

SKILLICORN, D., 2013. Other versions of the Enron data (preprocessed). [en línea]. [Consulta: 9 abril 2013]. Disponible en: <http://research.cs.queensu.ca/home/skill/otherforms.html>.

SLABY, A., 2007a. ROC analysis with Matlab. *Information Technology Interfaces, 2007. ITI 2007. 29th International Conference on* [en línea]. S.I.: IEEE, pp. 191–196. [Consulta: 11 octubre 2014]. Disponible en: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4283768.

SLABY, A., 2007b. ROC Matlab. [en línea]. [Consulta: 24 noviembre 2014]. Disponible en:

http://www.tech.plym.ac.uk/spmc/links/roc/roc_matlab.html.

SOKOLOVA, M., JAPKOWICZ, N. y SZPAKOWICZ, S., 2006. Beyond Accuracy, F-Score and ROC: A Family of Discriminant Measures for Performance Evaluation. *AI 2006: Advances in Artificial Intelligence* [en línea]. S.l.: Springer, Lecture Notes in Computer Science, pp. 1015-1021. [Consulta: 8 mayo 2015]. ISBN 978-3-540-49787-5. Disponible en: <https://www.aaai.org/Papers/Workshops/2006/WS-06-06/WS06-06-006.pdf>.

SPAM-ASSASSIN, 2013. Index of /publiccorpus. [en línea]. [Consulta: 9 abril 2013]. Disponible en: <http://spamassassin.apache.org/publiccorpus/>.

SPAMBAYES, 2013. SpamBayes: Bayesian anti-spam classifier written in Python. [en línea]. [Consulta: 27 noviembre 2012]. Disponible en: <http://spambayes.sourceforge.net/>.

SPAMMER-X, POSLUNS, J. y SJOUWERMAN, S., 2004. *Inside the SPAM Cartel*. S.l.: Syngress. ISBN 978-0-08-047850-0.

SPRENGERS, M. y HESKES, T.T., 2009. *The Effects of Different Bayesian Poison Methods on the Quality of the Bayesian Spam Filter 'SpamBayes'* [en línea]. S.l.: Radboud University Nijmegen. [Consulta: 28 enero 2015]. Disponible en: http://www.cs.ru.nl/bachelorscripties/2009/Martijn_Sprengers__0513288__The_Effects_of_Different_Bayesian_Poison_Methods_on_the_Quality_of_the_Bayesian_Spam_Filter_SpamBayes.pdf.

STRINGHINI, G., HOHLFELD, O., KRUEGEL, C. y VIGNA, G., 2014. The harvester, the botmaster, and the spammer: on the relations between the different actors in the spam landscape. [en línea]. S.l.: ACM Press, pp. 353-364. [Consulta: 3 julio 2015]. ISBN 978-1-4503-2800-5. DOI 10.1145/2590296.2590302. Disponible en: <http://dl.acm.org/citation.cfm?doid=2590296.2590302>.

STYLER, W., 2011. The EnronSent Corpus. *University of Colorado at Boulder Institute of Cognitive Science* [en línea], vol. Technical Report 01-2011. [Consulta: 1 julio 2015]. Disponible en: http://www.colorado.edu/ics/sites/default/files/attached-files/01-11_0.pdf.

SWETS, J.A., 1988. Measuring the accuracy of diagnostic systems. *Science*, no. 240, pp. 1285-1293.

SYSTAT SOFTWARE, 2015. *ROC Curves Analysis - SigmaPlot* [en línea]. 2015. S.l.: s.n. [Consulta: 17 febrero 2015]. Disponible en: http://www.sigmaplot.com/products/sigmaplot/ROC_Curves_Analysis.pdf.

TAPE, T.G. y UNIVERSITY OF NEBRASKA MEDICAL CENTER, 2014. The Area Under an ROC Curve. [en línea]. [Consulta: 28 junio 2015]. Disponible en: <http://gim.unmc.edu/dxtests/roc3.htm>.

THE APACHE SPAMASSASSIN PROJECT, 2013. SpamAssassin: combination of Bayesian filtering, header and text rules, whitelists, blacklists and collaborative filtering databases. [en línea]. [Consulta: 27 noviembre 2012]. Disponible en: <http://spamassassin.apache.org/>.

TREC, 2005. 2005 TREC Public Spam Corpus. [en línea]. [Consulta: 8 diciembre 2012]. Disponible en: <http://plg.uwaterloo.ca/~gvcormac/treccorpus/>.

TREC, 2006. 2006 TREC Public Spam Corpora. [en línea]. [Consulta: 8 diciembre 2012]. Disponible en: <http://plg.uwaterloo.ca/~gvcormac/treccorpus06/>.

- TREC, 2007. 2007 TREC Public Spam Corpus. [en línea]. [Consulta: 8 diciembre 2012]. Disponible en: <http://plg.uwaterloo.ca/~gvcormac/treccorpus07/>.
- TRUSTWAVE, 2013a. Spam Statistics (weekly updated) - Support. [en línea]. [Consulta: 6 diciembre 2012]. Disponible en: https://www.trustwave.com/support/labs/spam_statistics.asp.
- TRUSTWAVE, 2013b. Spam Types - Support. [en línea]. [Consulta: 20 diciembre 2013]. Disponible en: https://www.trustwave.com/support/labs/spam_types.asp.
- UEMURA, T., IKEDA, D. y ARIMURA, H., 2008. Unsupervised spam detection by document complexity estimation. *Discovery Science* [en línea]. S.l.: s.n., pp. 319–331. [Consulta: 10 marzo 2013]. Disponible en: <http://www.springerlink.com/index/u3p2846735751k11.pdf>.
- UNITED STATES SENATE y UNITED STATES HOUSE OF REPRESENTATIVES, 2003. Controlling the Assault of Non-Solicited Pornography And Marketing Act of 2003. [en línea], [Consulta: 27 enero 2015]. Disponible en: <http://www.legalarchiver.org/cs.htm>.
- UNSPAM TECHNOLOGIES, INC, 2004. Project Honey Pot. [en línea]. [Consulta: 2 julio 2015]. Disponible en: http://www.projecthoneypot.org/about_us.php.
- URIARTE, E.A. y MARTÍN, F.D., 2005. Topology preservation in SOM. *International Journal of Mathematical and Computer Sciences*, vol. 1, no. 1, pp. 19–22.
- VAPNIK, V.N., 1999. An overview of statistical learning theory. *Neural Networks, IEEE Transactions on*, vol. 10, no. 5, pp. 988–999.
- VESANTO, J., 1999. SOM-based data visualization methods. *Intelligent Data Analysis*, vol. 3, pp. 111–126.
- VESANTO, J., HIMBERG, J., ALHONIEMI, E. y PARHANKANGAS, J., 1999. Self-organizing map in Matlab: the SOM Toolbox. *Proceedings of the MATLAB DSP Conference* [en línea]. S.l.: s.n., pp. 16–17. [Consulta: 28 junio 2013]. Disponible en: http://cda.psych.uiuc.edu/matlab_class/martinez/edatoolbox/Docs/toolbox2paper.pdf.
- VESANTO, J., HIMBERG, J., ALHONIEMI, E. y PARHANKANGAS, J., 2000. SOM Toolbox for Matlab 5. [en línea]. S.l.: [Consulta: 14 julio 2013]. Disponible en: <http://www.cis.hut.fi/projects/somtoolbox/package/papers/techrep.pdf>.
- VILLMANN, T., DER, R., HERRMANN, M. y MARTINETZ, T., 1994. Topology preservation in self-organizing feature maps: general definition and efficient measurement. *Lecture Notes in Computer Science*, pp. 159–166.
- VILLMANN, T., DER, R. y MARTINETZ, T., 1994a. A new quantitative measure of topology preservation in Kohonen's feature maps. *Proceedings of the IEEE International Conference on Neural Networks* [en línea]. Orlando: s.n., pp. 645–648. [Consulta: 14 agosto 2014]. Disponible en: <http://www.inb.uni-luebeck.de/publications/pdfs/ViDeMa94.pdf>.
- VILLMANN, T., DER, R. y MARTINETZ, T., 1994b. A novel approach to measure the topology preservation of feature maps. *Proceedings of the International Conference on Artificial Neural Networks* [en línea]. S.l.: s.n., pp. 298–301. [Consulta: 14 agosto 2014]. Disponible en: <http://www.inb.uni-luebeck.de/publications/pdfs/ViDeMa94a.pdf>.

VRUSIAS, B.L. y GOLLEDGE, I., 2009a. Adaptable Text Filters and Unsupervised Neural Classifiers for Spam Detection. *Proceedings of the International Workshop on Computational Intelligence in Security for Information Systems CISIS'08* [en línea]. S.l.: s.n., pp. 195–202. [Consulta: 18 diciembre 2013]. Disponible en: extras.springer.com/2009/978-3-540-88180-3/papers/0053/00530195.pdf.

VRUSIAS, B.L. y GOLLEDGE, I., 2009b. Online Self-Organised Map Classifiers as Text Filters for Spam Email Detection. *Journal of Information Assurance and Security (JIAS)*, vol. 4, no. 2, pp. 151–160.

WANG, D., IRANI, D. y PU, C., 2013. A Study on Evolution of Email Spam Over Fifteen Years. *2013 9th International Conference Conference on Collaborative Computing: Networking, Applications and Worksharing (Collaboratecom)*. Atlanta, Georgia (USA): s.n., pp. 1-10.

WANG, D. y ZHANG, H., 2013. Inverse-Category-Frequency Based Supervised Term Weighting Schemes for Text Categorization. *Journal of Information Science & Engineering* [en línea], vol. 29, no. 2. [Consulta: 20 febrero 2014]. Disponible en: <http://search.ebscohost.com/login.aspx?direct=true&profile=ehost&scope=site&authtype=crawler&jrnl=10162364&AN=86233840&h=FsAsqRQCBYavGA%2FPgSc3j8hM18rr2VCzfi89ZBJh%2FhuVjzmzquqrWMKvurz0FMBojnTrVJuJfzp%2BSTLITPh6eJg%3D%3D&crl=c>.

WEISSTEIN, E.W., 2014. Plane. En: A plane is a two-dimensional doubly ruled surface spanned by two linearly independent vectors. The generalization of the plane to higher dimensions is called a hyperplane. The angle between two intersecting planes is known as the dihedral angle. The equation of a plane with nonzero normal vector $n=(a,b,c)$ through the point $x_0=(x_0,y_0,z_0)$ is $n \cdot (x-x_0)=0$, (1) where $x=(x,y,z)$. Plugging in gives the general equation of a plane, $ax+by+cz+d=0$, (2) where $d=-ax_0-by_0-cz_0$. (3) A... [en línea]. [Consulta: 26 septiembre 2014]. Disponible en: <http://mathworld.wolfram.com/Plane.html>.

WESTERLUND, M.L., 2005. Classification with Kohonen Self-Organizing Maps. [en línea], [Consulta: 5 julio 2013]. Disponible en: https://notendur.hi.is/benedikt/Courses/Mia_report2.pdf.

WIEHES, A., 2005. Comparing anti spam methods. [en línea], [Consulta: 26 septiembre 2014]. Disponible en: <http://brage.bibsys.no/xmlui/handle/11250/143763>.

WIKIPEDIA, 2012a. Anti-spam techniques. En: Page Version ID: 524163282, *Wikipedia, the free encyclopedia* [en línea]. [Consulta: 29 noviembre 2012]. Disponible en: http://en.wikipedia.org/w/index.php?title=Anti-spam_techniques&oldid=524163282.

WIKIPEDIA, 2012b. Image spam. En: Page Version ID: 521427814, *Wikipedia, the free encyclopedia* [en línea]. [Consulta: 6 diciembre 2012]. Disponible en: http://en.wikipedia.org/w/index.php?title=Image_spam&oldid=521427814.

WIKIPEDIA, 2013a. Bacn emails. [en línea]. [Consulta: 4 marzo 2013]. Disponible en: [http://en.wikipedia.org/wiki/Bacn_\(electronic\)](http://en.wikipedia.org/wiki/Bacn_(electronic)).

WIKIPEDIA, 2013b. Bayesian poisoning. [en línea]. [Consulta: 12 diciembre 2012]. Disponible en: http://en.wikipedia.org/wiki/Bayesian_poisoning.

WIKIPEDIA, 2013c. Binary classification. [en línea]. [Consulta: 14 abril 2013]. Disponible en: http://en.wikipedia.org/wiki/Binary_classification.

- WIKIPEDIA, 2013d. Enron Corpus. [en línea]. [Consulta: 9 abril 2013]. Disponible en: http://en.wikipedia.org/wiki/Enron_Corpus.
- WIKIPEDIA, 2013e. Greylisting. [en línea]. [Consulta: 8 diciembre 2012]. Disponible en: <http://en.wikipedia.org/wiki/Greylisting>.
- WIKIPEDIA, 2013f. Precision and recall. [en línea]. [Consulta: 14 abril 2013]. Disponible en: [http://en.wikipedia.org/wiki/Recall_\(information_retrieval\)](http://en.wikipedia.org/wiki/Recall_(information_retrieval)).
- WIKIPEDIA, 2013g. Receiver operating characteristic (ROC curve). [en línea]. [Consulta: 14 abril 2013]. Disponible en: http://en.wikipedia.org/wiki/Receiver_operating_characteristic.
- WIKIPEDIA, 2013h. Self-organizing map. *Wikipedia, the free encyclopedia* [en línea]. [Consulta: 25 mayo 2013]. Disponible en: http://en.wikipedia.org/w/index.php?title=Self-organizing_map&oldid=555704734.
- WIKIPEDIA, 2013i. Sensitivity and specificity. [en línea]. [Consulta: 14 abril 2013]. Disponible en: http://en.wikipedia.org/wiki/Sensitivity_and_specificity.
- WIKIPEDIA, 2014a. Accuracy and precision. [en línea]. [Consulta: 29 septiembre 2014]. Disponible en: http://en.wikipedia.org/wiki/Accuracy_and_precision.
- WIKIPEDIA, 2014b. Email. [en línea]. [Consulta: 24 enero 2015]. Disponible en: <http://en.wikipedia.org/wiki/Email#Origin>.
- WIKIPEDIA, 2014c. Pareto principle. [en línea]. [Consulta: 27 mayo 2014]. Disponible en: http://en.wikipedia.org/wiki/Pareto_principle.
- WILSON, D.R. y MARTINEZ, T.R., 2003. The general inefficiency of batch training for gradient descent learning. *Neural Networks*, vol. 16, no. 10, pp. 1429-1451. ISSN 08936080. DOI 10.1016/S0893-6080(03)00138-2.
- WITTEL, G.L. y WU, S.F., 2004. On Attacking Statistical Spam Filters. *CEAS* [en línea]. S.l.: s.n., [Consulta: 28 enero 2015]. Disponible en: http://pdf.aminer.org/000/085/123/on_attacking_statistical_spam_filters.pdf.
- WORSTER, A., FAN, J. y UPADHYE, S., 2006. Understanding receiver operating characteristic (ROC) curves. *Cjem*, vol. 8, no. 1, pp. 19–20.
- YERAZUNIS, W., KATO, M., KORI, M., SHIBATA, H. y HACKENBERG, K., 2010. Keeping the Good Stuff In: Confidential Information Firewalling with the CRM114 Spam Filter & Text Classifier. *Black Hat* [en línea]. USA: s.n., [Consulta: 28 enero 2015]. Disponible en: <http://www.merl.com/publications/docs/TR2010-064.pdf>.
- ZEIMPEKIS, D., KONTOPOULOU, E.M. y GALLOPOULOS, E., 2011. Text to Matrix Generator (TMG). [en línea]. [Consulta: 20 diciembre 2013]. Disponible en: <http://scgroup20.ceid.upatras.gr:8000/tmg/>.
- ZHANG, Y., 2012. Lecture for Chapter 2 - Data Preprocessing. [en línea]. S.l. [Consulta: 11 marzo 2013]. Disponible en: <http://www.cs.gsu.edu/~cscyzq/courses/dm/slides/ch02.ppt>.
- ZHOU, D., BOUSQUET, O., LAL, T.N., WESTON, J. y SCHÖLKOPF, B., 2004. Learning with

local and global consistency. *Advances in neural information processing systems*, vol. 16, pp. 321–328 (8).

ZHOU, D., BURGESS, C.J.C. y TAO, T., 2007. Transductive link spam detection. *Proceedings of the 3rd international workshop on Adversarial information retrieval on the web* [en línea]. S.l.: s.n., pp. 21–28 (8). [Consulta: 11 diciembre 2012]. Disponible en: <http://dl.acm.org/citation.cfm?id=1244413>.

ZHU, X., 2008. Semi-supervised learning literature survey. , pp. 60.

ZHU, X., 2010. Semi-supervised learning. *Encyclopedia of Machine Learning* [en línea]. S.l.: Springer, pp. 892–897. [Consulta: 29 enero 2015]. Disponible en: http://link.springer.com/content/pdf/10.1007/978-0-387-30164-8_749.pdf.

ZIEN, A., 2006. Semi-Supervised Support Vector Machines and Application to Spam Filtering (PPT). [en línea]. S.l.: s.n., pp. 37. [Consulta: 29 noviembre 2012]. Disponible en: http://www.is.tuebingen.mpg.de/fileadmin/user_upload/files/publications/SSL-spam_4162%5B0%5D.pdf.

ZWEIG, M.H. y CAMPBELL, G., 1993. Receiver-operating characteristic (ROC) plots: a fundamental evaluation tool in clinical medicine. *Clinical Chemistry*, vol. 39, no. 4, pp. 561–577.