INSTITUTO POLITECNICO NACIONAL

CENTRO DE INVESTIGACIÓN EN COMPUTACIÓN

MÁQUINAS ASOCIATIVAS ALFA-BETA CON SOPORTE VECTORIAL

T E S I S

QUE PARA OBTENER EL GRADO DE

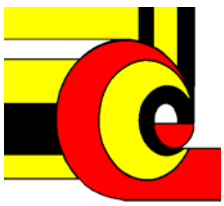
DOCTOR EN CIENCIAS DE LA COMPUTACÓN

PRESENTA:

LUIS OCTAVIO LÓPEZ LEYVA

DIRECTORES DE TESIS:

DR. CORNELIO YÁÑEZ MÁRQUEZ
DR. OSCAR CAMACHO NIETO



MÉXICO, D.F. NOVIEMBRE DE 2008

AGRADECIMIENTOS

Para mis asesores *Dr. Cornelio Yáñez Márquez* y *Dr. Oscar Camacho Nieto*, por brindarme el apoyo incondicional en todo momento en el desarrollo de la tesis doctoral. También quiero expresar de una manera muy especial al compañero, amigo y hermano muchas gracias Cornelio, por ayudarme a cruzar este difícil camino. Se que eres un científico de excelencia y de una nobleza extraordinaria.

A mis sinodales *Dr. Sergio Suárez Guerra* y *Dr. Lindig Bos*, a ellos en especial se debe la idea del método denominado (*Suárez-Lindig*), que sirve para obtener automáticamente un valor umbral para el proceso de binarización de las imágenes en escala de grises que conforman las bases de datos MNIST. Al Dr. *Lindig* se le ocurrió una ventana de 3x3 pixeles, y que esta ventana tomara la información de los 8 vecinos contiguos al píxel central, y al Dr. *Suárez* desarrolló la fórmula matemática. Con esto método se logró reducir el error grandemente hasta en 1.54%.

Al Dr. *Oleksiy Pogrebnyak* gracias por sus críticas constructivas y excelentes aportaciones vertidas en este trabajo, sus oportunos comentarios ayudaron a mejorar esta tesis.

Al Dr. Lindig Bos, por su valiosa intervención en el desarrollo de la complejidad del algoritmo.

Dr. *Amadeo Argüelles Cruz*, gracias amigo por ofrecerme tu ayuda incondicional y recibir tu apoyo en la elaboración de los diagramas de flujo del algoritmo.

M. en C: *Itzamá López Yánez*, gracias por apoyo recibido por la elaboración final del documento de tesis.

M. en C. *Mario Aldape Pérez*, gracias por el apoyo en la elaboración de la presentación del tema de tesis, y tus atinados comentarios de aliento.

M. en C. *Rolando Flores Carapia*, por permitirme ser tu amigo y brindarme el apoyo en la realización del software del algoritmo que sustenta esta tesis.

ING: *Javier Marín García*, por todo el apoyo antes y durante el desarrollo de la tesis doctoral.

C. M. en C. *Flor Susana Rodríguez Mora*, gracias por tu apoyo incondicional.

M. en C. Benjamín Luna Venoso gracias por tu valiosa ayuda en matemáticas.

A los integrantes del grupo *Alfa-Beta*, por su apoyo y que gracias a sus comentarios e intercambio de ideas, generan siempre una atmósfera propicia para la investigación

A *COTEPABE*, por que gracias a su respaldo a la investigación realizada por un servidor, se ha logrado la meta de concluir el posgrado en tiempo y forma.

Al *CIC* y al *IPN*, a las que me debo como profesionista y por las que siempre estaré al pendiente de poner en alto su nombre.

Síntesis del Resumen

Las Máquinas de Soporte Vectorial son una clase muy específica de algoritmos, que se caracterizan por la utilización de kernels, la ausencia de los mínimos locales, la poca densidad de la solución y el control de la capacidad obtenido por actuar en el margen, o en otra "dimensión independiente", cantidades tales como el número de vectores de soporte. Fueron inventados por Boser, Guyon y Vladimir Vapnik, y se introdujo por primera vez en una ponencia; COLT 1992. Todas estas características, sin embargo, ya estaban presentes y se ha utilizado en la máquina de aprendizaje desde el década de 1960: hyperplanes de máximo margen en el espacio de entrada fueron discutidos por, Duda y Hart, Cover, Vapnik y colaboradores, y varios artículos de la mecánica estadística (por ejemplo Anlauf y Biehl); la utilización de kernels fue propuesta por Aronszajn, Wahba, Poggio, y otros, pero es el artículo de Aizermann y colaboradores, publicado en 1964 que introdujo la interpretación geométrica de los kernels como productos internos en un espacio de características. Técnicas de optimización similares fueron utilizadas en reconocimiento de patrones por Mangasarian, y la poca densidad también había sido ya discutida. El uso de variables de holgura para superar el problema de ruido y la no de separabilidad fue introducida también en la década de 1960 por Smith y mejorado por Bennett y Mangasarian. Sin embargo, no fue hasta 1992 que todas esas características fueron puestas juntas para formar el clasificador de máximo margen, la máquina básica de soporte vectorial, y no fue hasta 1995 que la versión del margen suave fue introducido: presentado por Cortés y Vapnik es sorprendente cómo, naturalmente, y elegantemente todas las piezas encajan y se complementan entre sí. Los artículos de Shawe-Taylor y colaboradores Bartlett dio el primer límite estadístico riguroso sobre la generalización de la margen duro de las SVMs, mientras que el artículo de Shawe-Taylor y Cristianini ofrece límites similares para los algoritmos de margen suave y para el caso de regresión.

Después de su introducción, un número creciente de investigadores que han trabajado en ambos algorítmos y el análisis teórico de esos sistemas, creando en pocos años lo que es efectivamente una nueva dirección de investigación por derecho propio, emergiendo conceptos de disciplinas tan distantes como: estadísticas, análisis funcional, optimización, así como la máquina de aprendizaje. El clasificador de margen suave fue introducido unos años más tarde por Vapnik y Cortés, y en 1995 el algoritmo fue extendido al caso de regresión.

Los dos últimos libros escritos por Vapnik proveen una muy amplia base teórica de la materia y desarrolla los conceptos de una Máquina de Soporte Vectorial.

El algoritmo ν- vector soporte para la clasificación y regresión fue introducido y desarrollado por Smola, Schoelkopf, Williamson y Bartlett. Otras adaptaciones de los planteamientos básicos se han utilizado para la estimación de la densidad, la transducción, la estimación de puntos de Bayes, la regresión ordinal.

Recientemente, una serie de aplicaciones prácticas de SVMs han sido reportadas en campos tan diversos como la bioinformática, la lingüística computacional y la visión por computadoras. Muchos de estos avances recientes son presentados en las colecciones Schoelkopf, Smola y colaboradores, por Burgues, Smola, Schoelkopf y por Evgeniou, Pontil y Poggio.

Por último, la tesis de doctorado de Cortés, Osuna, Schölkopf y Smola proporcionar una valiosa fuente de primera mano de temas como el trabajo de investigación sobre la viabilidad en esta brecha.

En este trabajo de tesis se presenta un nuevo modelo de reconocimiento automático de patrones, las Máquinas Asociativas Alfa-Beta con Soporte Vectorial, mismo que se ubica dentro del Enfoque Asociativo de Reconocimiento de Patrones. Este modelo nace al fusionar algunos elementos teóricos de las memorias asociativas Alfa-Beta con algunos elementos tomados de las *Support Vector Machines*, originando a su vez varias transformadas matemáticas novedosas.

Además, se presenta un estudio experimental del desempeño del algoritmo propuesto, comparando su rendimiento de clasificación con el exhibido por otros reconocedores de patrones, al trabajar con diversas bases de datos de acceso público.

En dichos experimentos se muestra que el modelo propuesto exhibe resultados competitivos con respecto a algunos de los algoritmos más ampliamente conocidos, presentes en la literatura científica contemporánea, al aplicarse al reconocimiento de dígitos escritos a mano; en particular de la base de datos MNIST.

Con este trabajo de tesis se engrosan las filas del Enfoque Asociativo de Clasificación de Patrones, con un modelo de alto desempeño que ofrece una eficacia competitiva, además de ser robusto ante patrones modificados con alteraciones mezcladas.

Índice General

Índice de tablas xi
Índice de figuras xii

Capítulo 1: Introducción 1

1.1 Antecedentes 1
1.2 Objetivo 4
1.3 Aportaciones 4
1.4 Organización del documento 5

Capítulo 2: Estado del arte 6

2.1 Memorias asociativas 6
2.2 Operadores α y β 7
2.3 Support Vector Machines 9

Capítulo 3: Modelo propuesto 11

3.1 Descripción de las máquinas asociativas Alfa beta con Soporte Vectorial- ejemplo gráfico 12
3.2 Definición y resultados preliminares 18
3.3 El vector soporte para el nuevo modelo 42
3.3.1 Ejemplos ilustrativos binarias α y β 42
3.3.2 Análisis de los ejemplos ilustrativos 48
3.3.3 Definición del nuevo vector soporte, ejemplos y resultados 53
3.4 Dos nuevas transformadas 62
3.5 Algoritmo principal 65
3.6 Complejidad del algoritmo principal 66

Capítulo 4: Resultados y Discusión 73

4.1 Base de datos MNIST de dígitos escritos a mano 73
4.1.1 Las diez clases de la base de datos MNIST 74
4.1.2 Dos experimentos sin preprocesamiento 76
4.1.3 Método de umbralización basado en promedios locales 77
4.1.4 Método de umbralización de Otsu 81
4.1.5 Método de umbralización basado en histogramas 82
4.1.6 Tabla de resultados 86
4.2 Base de datos Iris Plant 86

Capítulo 5: Conclusiones y trabajo futuro 89

5.1 Conclusiones 89
5.2 Trabajo futuro 89

Publicaciones **91**

Apéndice A – Diagrama de flujo del algoritmo **93**

Figura A.1 Diagrama de flujo para la fase de aprendizaje 94
Figura A.2 Diagrama de flujo para la fase de recuperación – 1ª. Parte 95
Figura A.3 Diagrama de flujo para la fase de recuperación – 2ª. Parte 96
Figura A.4 Diagrama de flujo para la fase de recuperación – 3ª. Parte 97

Referencias 98

Índice de tablas

Tabla	*Descripción*	*Página*
2.1	*Definición de los operadores* Alfa y Beta	8
2.2	*Propiedades de la operación binaria* α	8
2.3	*Propiedades de la operación binaria* β	8
2.4	*Propiedades de la aplicación combinada de las operaciones* α y β	8
4.1	*Comparación entre varios algoritmos aplicados a la base de datos* ***MNIST***.	86
4.3	*Rasgos de la base de datos* ***Iris Plant***	87
4.4	*Comparación del rendimiento con la base de datos* ***Iris Plant***	87

Índice de figuras

Figura	*Descripción*	*Página*
2.1	*Support Vector Machine*	9
2.2	*Ejemplo de función Kernel*	10
3.1	*Conjunto fundamental*	12
3.2	*Patrón con la información repetida (vector de soporte)*	13
3.3	*Conjunto fundamental con la información presente en todos los patrones fundamentales eliminada*	13
3.4	*Conjunto fundamental negado*	13
3.5	*Patrón con la información repetida, ausente en todos los patrones fundamentales*	14
3.6	*Conjunto fundamental con la información ausente en todos los patrones fundamentales eliminada*	14
3.7	*Patrón desconocido*	14
3.8	*Patrón desconocido al que se ha eliminado la información presente repetida*	14
3.9	*Patrón desconocido negado*	15
3.10	*Patrón desconocido al que se ha eliminado la información ausente repetida*	15
3.11	*Patrón fundamental sin la información presente repetida menos diferente al patrón desconocido al que se ha eliminado la información presente repetida*	15
3.12	*Patrón fundamental sin la información ausente repetida menos diferente al patrón desconocido al que se ha eliminado la información ausente repetida.*	16
3.13	*Patrón de salida*	16
3.14	*Recuperación del segundo patrón fundamental*	16
3.15	*Recuperación del tercero patrón fundamental*	17
3.16	*Recuperación del cuarto patrón fundamental*	17
3.17	*Recuperación del quinto patrón fundamental*	17
4.1	*Patrones de entrenamiento para la clase 0*	74
4.2	*Patrones de entrenamiento para la clase 1*	74
4.3	*Patrones de entrenamiento para la clase 2*	75
4.4	*Patrones de entrenamiento para la clase 9*	75
4.5	*Conjunto de entrenamiento.*	75
4.6	*Ventana de 10 instancias de entrenamiento*	76
4.7	*Ventana de 60 instancias de prueba*	76
4.8	*Imagen a ser binarizada.Un píxel blanco representa un valor alto, mientras que un píxel azul representa un valor bajo, ambos en la escala de grises*	78
4.9	*Ventana de 3X3*	78
4.10	*Ventana centrada en el píxel (6, 4) de la imagen*	79
4.11	*Caso de interés:píxel del fondo rodeado de pixeles de fondo .*	79
4.12	*Caso de interés: píxel del objeto rodeado de pixeles de objeto*	80
4.13	*Caso de interés: píxel de la orilla de la imagen*	80
4.14	*Resultado de la binarización*	81
4.15	*Base de datos MNIST*	85

Introducción

En este trabajo de tesis se presentan las máquinas asociativas alfa-beta con soporte vectorial, que conforman un nuevo modelo de reconocimiento automático de patrones, el cual surge al tomar elementos de dos ramas importantes del reconocimiento de patrones. Por un lado, se utiliza el modelo de las memorias asociativas Alfa-Beta como punto de partida, tomando de éste los operadores Alfa y Beta y sus propiedades; y por otro, la idea del vector de soporte se toma de la teoría de las máquinas de soporte vectorial (SVM, por sus siglas en inglés). El nuevo modelo exhibe un desempeño experimental competitivo, al ser comparado con otros importantes clasificadores descritos en la literatura actual.

Resumen

En este trabajo de tesis se presenta un nuevo modelo de reconocimiento automático de patrones, las Máquinas Asociativas Alfa-Beta con Soporte Vectorial, mismo que se ubica dentro del Enfoque Asociativo de Reconocimiento de Patrones. Este modelo nace al fusionar algunos elementos teóricos de las memorias asociativas Alfa-Beta con algunos elementos tomados de las *Support Vector Machines*, originando a su vez varias transformadas matemáticas novedosas.

Además, se presenta un estudio experimental del desempeño del algoritmo propuesto, comparando su rendimiento de clasificación con el exhibido por otros reconocedores de patrones, al trabajar con diversas bases de datos de acceso público.

En dichos experimentos se muestra que el modelo propuesto exhibe resultados competitivos con respecto a algunos de los algoritmos más ampliamente conocidos, presentes en la literatura científica contemporánea, al aplicarse al reconocimiento de dígitos escritos a mano; en particular de la base de datos MNIST.

Con este trabajo de tesis se engrosan las filas del Enfoque Asociativo de Clasificación de Patrones, con un modelo de alto desempeño que ofrece una eficacia competitiva, además de ser robusto ante patrones modificados con alteraciones mezcladas.

Abstract

In the current document of thesis, a new model for automatic pattern recognition is presented. This new model, the Alpha-Beta Associative Support Vector Machines, is a member of the Associative Approach of Pattern Recognition. This model arises when some theoretical elements from the Alpha-Beta associative memories are merged with some theoretical elements taken from Support Vector Machines, giving also birth to several new mathematical transforms.

An experimental study of the proposed algorithm performance is presented. In this study, the classification performance of the Alpha-Beta Associative Support Vector Machines is compared to that exhibited by other classifiers, while working with different data bases of public domain.

In particular, when applied to handwritten digits recognition (namely in the MNIST database) the Alpha-Beta Associative Support Vector Machines exhibit competitive results against some of the most widely known algorithms currently available in scientific literature.

With this work of thesis, the number of models belonging to the Associative Approach of Pattern Recognition has been increased, with a high performance model which offers a very competitive efficacy, besides being robust against patterns modified with mixed alterations.

Capítulo 1

Introducción

En este trabajo de tesis se presentan las máquinas asociativas alfa-beta con soporte vectorial, que conforman un nuevo modelo de reconocimiento automático de patrones, el cual surge al tomar elementos de dos ramas importantes del reconocimiento de patrones. Por un lado, se utiliza el modelo de las memorias asociativas Alfa-Beta como punto de partida, tomando de éste los operadores α y β y sus propiedades; y por otro, la idea del vector de soporte se toma de la teoría de las máquinas de soporte vectorial (SVM, por sus siglas en inglés). El nuevo modelo exhibe un desempeño experimental competitivo, al ser comparado con otros importantes clasificadores descritos en la literatura actual.

1.1. Antecedentes

El proceso de reconocer cosas, fenómenos y conceptos, tales como un sonido, cierta comida, un depredador, una imagen, un amigo, un sabor, entre otras instancias, es algo que los seres vivos hacemos de forma inconsciente, pero que nos permite adaptarnos a nuestro entorno y puede ser clave en el momento de la supervivencia [1]. La gran importancia de las tareas de reconocimiento de patrones realizadas cotidianamente, ha conducido a que en ciertos equipos de investigación científica, sobre todo con el advenimiento de los sistemas computacionales modernos, surja la idea de crear, diseñar e implementar sistemas de reconocimiento automático de patrones [2].

Una de las primeras observaciones de quienes nos dedicamos a tratar de resolver este tipo de problemas en una computadora, es que la solución general es muy complicada, dado que son muchos los factores que están involucrados en el proceso de reconocer. Por ello, la identificación o selección de rasgos o características en los objetos, procesos, fenómenos y conceptos a reconocer de manera automática, es una actividad que se realiza de manera inductiva, de lo simple a lo complejo, de lo concreto a lo abstracto, y con una buena dosis de ensayo y error [3]. No obstante, es preciso aclarar que la selección de rasgos en los sistemas modernos automáticos de reconocimiento de patrones es una línea de investigación vigente [4].

Después de haber seleccionado los rasgos o características de las instancias en estudio, se crean vectores que representan a esas instancias, cada una de cuyas componentes es uno de dichos rasgos o características: a esos vectores se les conoce como

patrones, aunque por convención en la literatura científica el término patrón se refiere de manera indistinta a la instancia o al vector que la representa. Acto seguido, se requiere del diseño e implementación de una estrategia que permita crear y operar un sistema computacional que automáticamente reconozca patrones. Esta es la etapa de reconocimiento de patrones, la cual es posible sólo si existió previamente una etapa de aprendizaje o entrenamiento en el sistema automático [5], [6].

Dependiendo de la aplicación y del problema específico, la fase de reconocimiento de los patrones en el sistema automático puede operarse con al menos dos intenciones: recuperar o clasificar los patrones en estudio [2], [3]. Por ello, gran parte de la literatura relacionada con el área de reconocimiento de patrones menciona explícitamente la clasificación de patrones [7], [8].

Actualmente, entre los principales enfoques de Reconocimiento Automático de Patrones, se encuentran los siguientes:

- *Enfoque estadístico-probabilístico*. Su principal clasificador está basado en la teoría de la probabilidad, y específicamente en el teorema de Bayes. Es históricamente el primer enfoque que existió y probablemente el más desarrollado [3], [5], [9]-[17].

- *Clasificadores basados en métricas*. Usualmente se ubican dentro del enfoque estadístico y se basan en el concepto de métrica y en las propiedades de los espacios métricos para hacer la clasificación [18]-[29].

- *Enfoque sintáctico-estructural*. Se basa en la teoría de autómatas y lenguajes formales para hacer la clasificación. Se enfoca más en la estructura de las cosas a clasificar que en mediciones numéricas [9], [12], [30], [31].

- *Enfoque neuronal*. Se basa en modelos matemáticos de las neuronas del cerebro humano y, a diferencia del enfoque estadístico-probabilístico, los sistemas automáticos de reconocimiento de patrones basados en en el enfoque neuronal, además de clasificar patrones, también son capaces de recuperarlos [9]-[13], [32]-[51].

- *Enfoque asociativo*. Este enfoque fue creado en el Centro de Investigación en Computación del IPN en 2002, y utiliza los modelos de memorias asociativas para diseñar e implementar reconocedores de patrones robustos ante la presencia de patrones alterados. Los sistemas automáticos de reconocimiento de patrones basados en en el enfoque asociativo son capaces de realizar la tarea de clasificación de patrones, como un caso particular de la tarea principal que realizan de manera eficiente: recuperar patrones [4], [7], [51]-[72]

En cada uno de los enfoques mencionados, se han creado algoritmos para diseñar reconocedores de patrones. Por ejemplo, el reconocedor de patrones más famoso en el enfoque estadístico-probabilístico es el clasificador bayesiano [3], y el clasificador euclidiano se identifica con el enfoque basado en métricas [73]. Sin embargo, la mayoría de los algoritmos de reconocimiento de patrones comparten características de más de un enfoque y elementos de otras áreas de la ciencia, especialmente de disciplinas

matemáticas. Esto es particularmente notorio en el clasificador euclidiano dado que, no obstante que es un clasificador basado en métricas, también utiliza de manera importante la teoría de las funciones discriminantes, misma que no interviene en otros clasificadores basados en métricas. Esta situación de tomar elementos de un enfoque específico de reconocimiento de patrones y algunas disciplinas matemáticas sucede también con las Máquinas de Soporte Vectorial (en la literatura científica actual, se refiere a este modelo por sus siglas en inglés, SVM, por lo que en el presente documento continuaremos con esta tendencia), técnica desarrollada por Vapnik que utiliza un algoritmo de entrenamiento, el cual maximiza el margen entre los patrones en el límite de clases; estos patrones han sido llamados por Vapnik *vectores de soporte* (support vectors) [74], [75]. Cabe mencionar que esta técnica ha sido aplicada exitosamente en diversas áreas de la actividad humana [76]-[79].

Entre los algoritmos reconocedores de patrones existentes en la actualidad que pertenecen a los enfoques anteriores, hay dos que sobresalen porque comparten notables características deseables en un algoritmo de este tipo, a saber: su sencillez y su alta eficacia.

Uno es el clasificador k-NN (k-nearest neighbor, los k-vecinos más cercanos), el cual es un modelo de mínima distancia y se ubica en los clasificadores basados en métricas. El método del k-NN calcula la distancia de un patrón de prueba respecto a cada uno de los patrones de aprendizaje o entrenamiento, ordena las distancias de menor a mayor y retiene la clase que se obtiene por mayoría entre los k patrones más cercanos [18], [27], [80], [81]. En el clásico proyecto Statlog, desarrollado en Escocia por Michie y sus colaboradores a inicios de la década de los noventa, se muestra experimentalmente la superioridad del k-NN respecto de un buen número de clasificadores automáticos de patrones [13]. Sin embargo, la gran desventaja del k-NN es su poca eficiencia, puesto que cuando se trabaja con un conjunto amplio de patrones de aprendizaje, el hecho de calcular las distancias de todos esos patrones con respecto al patrón de prueba y, posteriormente, ordenar las distancias, es un proceso computacionalmente caro [18], [82]-[84].

El otro enfoque es el asociativo, mismo que se inició con dos trabajos de tesis desarrollados durante 2002 en el CIC-IPN: una tesis de doctorado en ciencias de la computación [53], donde se introdujeron los operadores α y β, y una tesis de maestría en ciencias de la computación [52], la cual combina dos importantes modelos de memorias asociativas para generar un clasificador eficiente. El enfoque asociativo de reconocimiento de patrones se basa en las memorias asociativas, cuyos modelos matemáticos pioneros [85] son contemporáneos a los primeros modelos de redes neuronales [33]. El estado del arte en las memorias asociativas que sirven de base a modelos asociativos de reconocimiento de patrones, está constituido por las memorias asociativas Alfa-Beta, cuyo modelo se basa en dos operaciones simples, α y β, las cuales son equiparables en sencillez a las operaciones básicas de la lógica booleana [53]. En un número importante de investigaciones recientes, se ha mostrado teórica y experimentalmente que, a pesar de su sencillez, los modelos de reconocimeinto automático de patrones basados en las memorias asociativas Alfa-Beta son altamente eficaces, así como competitivos con los modelos reportados en la literatura actual [4], [7], [8], [51]-[68], [71], [72], [87]. Además, estos modelos son más eficientes que el k-NN para conjuntos

grandes de patrones de aprendizaje [86].

Cabe mencionar que las memorias asociativas Alfa-Beta trabajan solamente con patrones binarios, por lo que dichos patrones pueden presentar alteraciones aditivas, sustractivas o mezcladas. En este sentido, existen dos tipos de memorias asociativas Alfa-Beta: uno que es muy robusto ante alteraciones aditivas pero muy sensible ante alteraciones sustractivas, mientras que el otro tipo es muy robusto ante alteraciones sustractivas pero muy sensible ante alteraciones aditivas. La gran desventaja que presentan estos modelos matemáticos, es que la presencia de alteraciones mezcladas afecta fuertemente y de manera negativa su eficacia, puesto que son sensibles ante ellas [53], [54], [56], [87]. El problema ha sido parcialmente resuelto como resultado de algunos trabajos de investigación desarrollados por miembros del Grupo Alfa-Beta, quienes han ideado la aplicación de conceptos y técnicas de diversas áreas, a saber: el desarrollo y uso del código Johnson-Möbius modificado [56], la creación y aplicación de las redes neuronales Alfa-Beta sin pesos [51], la creación, fundamentación y aplicación de las memorias asociativas bidireccionales Alfa-Beta [54], la creación y uso de las multimemorias Alfa-Beta [65] y el desarrollo y aplicación del clasificador Gama [67]; no obstante los esfuerzos realizados, este problema permanece sin una solución definitiva, y con el presente trabajo de tesis se pretende avanzar en esa dirección.

De esta manera, la presente propuesta de tema de tesis queda delimitida dentro del área de investigación del enfoque asociativo de reconocimiento automático de patrones, en sus tareas de recuperación y clasificación de patrones.

1.2. Objetivo

Crear e implementar un modelo matemático de reconocimiento automático de patrones, las máquinas asociativas Alfa-Beta con soporte vectorial, que surja al tomar elementos de dos ramas importantes del reconocimiento de patrones. Por un lado, los operadores α y β y sus propiedades, aportación del modelo de las memorias asociativas Alfa-Beta; y por otro, la idea del vector de soporte, que proviene de la teoría de las máquinas de soporte vectorial (SVM, por sus siglas en inglés). El nuevo modelo deberá exhibir un desempeño experimental competitivo, al ser comparado con otros importantes clasificadores descritos en la literatura actual.

1.3. Aportaciones

- Una transformada vectorial original, llamada Transformada $\boldsymbol{\tau}$, cuyos argumentos de entrada son dos vectores de igual dimensión con componentes binarias, que permite conocer la cantidad y ubicación de la alteración aditiva del segundo vector de entrada respecto del primero.
- Un nuevo modelo de reconocimiento automático de patrones llamado máquinas asociativas Alfa-Beta con soporte vectorial, que exhibe un desempeño experimental competitivo, al ser comparado con otros importantes clasificadores descritos en la literatura actual.
- Análisis experimental del nuevo modelo, al aplicarlo en bases de datos conocidas.

1.4. Organización del documento

En este Capítulo se han presentado: los antecedentes, el objetivo y las aportaciones de este trabajo de tesis . El resto del documento está organizado de la siguiente manera:

En el Capítulo 2 se presenta el estado del arte de las dos áreas de investigación que sirven de base para la presente tesis, que son las memorias asociativas Alfa-Beta por un lado, y las Máquinas de Soporte Vectorial por otro.

A su vez, el Capítulo 3 es la parte más relevante de este documento. En el mismo se introduce el algoritmo propuesto, junto con las operaciones y herramientas matemáticas desarrolladas en este trabajo de tesis, que le dan sustento. El contenido del Capítulo incluye las definiciones de las herramientas que sustentan el nuevo modelo, así como el desarrollo del algoritmo principal, cuyo diagrama de flujo constituye el Apéndice A.

Los resultados experimentales, así como la discusión de los mismos, se presentan en el Capítulo 4; y en el Capítulo final, el 5, se exponen las conclusiones y recomendaciones para trabajo futuro. Finalmente, se incluyen las referencias bibliográficas.

Capítulo 2

Estado del Arte

En este capítulo se presenta el estado del arte de las dos áreas de investigación que sirven de base para la presente tesis, que son las memorias asociativas Alfa-Beta por un lado, y las Máquinas de Soporte Vectorial (SVM) por otro.

En la sección 2.1 se describen brevemente los conceptos básicos de memorias asociativas, y se presentan los operadores operadores α y β y sus propiedades; es muy importante hacer notar que de las memorias asociativas $\alpha\beta$ se toman **sólo** los operadores operadores α y β y sus propiedades, no así los algoritmos de las fases de aprendizaje y recuperación de esas memorias.

En la sección 2.2 se describen de manera breve las máquinas de soporte vectorial (SVM) y se ejemplifica gráficamente su funcionamiento; es preciso aclarar que de las SVM **sólo** se tomará la idea del vector de soporte, no así su concepto, puesto que el concepto del nuevo vector de soporte es totalmente diferente del concepto que usan las SVM clásicas. A fin de ilustrar las enormes diferencias que existen entre las SVM clásicas y las máquinas asociativas Alfa-Beta con soporte vectorial, se plantea el hecho de que,mientras que las SVM clásicas trabajan en modo biclase, el nuevo modelo de máquinas asociativas Alfa-Beta con soporte vectorial es capaz de resolver problemas multiclase.

2.1. Memorias Asociativas

Los conceptos que a continuación se presentan están fuertemente basados en [53], [54], [69], [70], [88].

Una Memoria Asociativa puede formularse, para su operación, como un sistema de entrada y salida, idea que se esquematiza a continuación.

$$\mathbf{x} \longrightarrow \boxed{\mathbf{M}} \longrightarrow \mathbf{y}$$

En este esquema, los patrones de entrada y salida están representados por vectores columna denotados por $\mathbf{x}$ y $\mathbf{y}$, respectivamente. Cada uno de los patrones de entrada forma una asociación con el correspondiente patrón de salida, la cual es similar a una pareja ordenada; por ejemplo, los patrones $\mathbf{x}$ y $\mathbf{y}$ del esquema anterior forman la asociación $(\mathbf{x}, \mathbf{y})$. A un patrón de entrada $\mathbf{x}^1$ le corresponderá el patrón de salida

$\mathbf{y}^1$, y ambos formarán la asociación $(\mathbf{x}^1, \mathbf{y}^1)$; del mismo modo, para un número entero positivo k específico, la asociación correspondiente es $(\mathbf{x}^k, \mathbf{y}^k)$.

La memoria asociativa $\mathbf{M}$ se representa mediante una matriz, la cual se genera a partir de un conjunto finito de asociaciones conocidas de antemano: este es el conjunto fundamental de aprendizaje, o simplemente conjunto fundamental. El conjunto fundamental se representa de la siguiente manera: $\{(\mathbf{x}^\mu, \mathbf{y}^\mu) \mid \mu = 1, 2, ..., p\}$ donde p es un número entero positivo que representa la cardinalidad del conjunto fundamental. La naturaleza del conjunto fundamental proporciona un importante criterio para clasificar las memorias asociativas: Una memoria es autoasociativa si se cumple que $\mathbf{x}^\mu = \mathbf{y}^\mu$ $\forall \mu \in \{1, 2, ..., p\}$, por lo que uno de los requisitos que se debe de cumplir es que $n = m$. Por otro lado, una memoria heteroasociativa es aquella en donde $\exists \mu \in \{1, 2, \ldots, p\}$ para el que se cumple que $\mathbf{x}^\mu \neq \mathbf{y}^\mu$. Nótese que puede haber memorias heteroasociativas con $n = m$.

En los problemas donde intervienen las memorias asociativas, se consideran dos fases importantes: La fase de aprendizaje, que es donde se genera la memoria asociativa a partir de las p asociaciones del conjunto fundamental, y la fase de recuperación u operación que es donde la memoria asociativa opera sobre un patrón de entrada con objeto de reconocerlo, a la manera del esquema que aparece al inicio de esta subsección.

A fin de especificar las componentes de los patrones, se requiere la notación para dos conjuntos a los que llamaremos arbitrariamente A y B. Las componentes de los vectores columna que representan a los patrones, tanto de entrada como de salida, serán elementos del conjunto A, y las entradas de la matriz $\mathbf{M}$ serán elementos del conjunto B. No hay requisitos previos ni limitaciones respecto de la elección de estos dos conjuntos, por lo que no necesariamente deben ser diferentes o poseer características especiales.

Por convención, cada vector columna que representa a un patrón de entrada tendrá n componentes cuyos valores pertenecen al conjunto A, y cada vector columna que representa a un patrón de salida tendrá m componentes cuyos valores pertenecen también al conjunto A; es decir: $\mathbf{x}^\mu \in A^n$ y $\mathbf{y}^\mu \in A^m$ $\forall \mu \in \{1, 2, ..., p\}$. La j-ésima componente de un vector columna se indicará con la misma letra del vector, pero sin negrilla, colocando a j como subíndice: la j-ésima componente del vector columna $\mathbf{x}^\mu$ se representa por: $\mathbf{x}^\mu_j$.

2.2. Operadores α y β

Las memorias asociativas Alfa-Beta, por su parte, son de dos tipos y pueden operar en dos modos diferentes. El operador α es utilizado en la fase de aprendizaje, mientras que el operador β es útil durante la fase de recuperación. Estos dos operadores fueron definidos de manera tabular y sus propiedades demostradas en [53]; a continuación se incluyen las tablas que representan a los operadores α y β,: dados los conjuntos $A = \{0, 1\}$ y $B = \{0, 1, 2\}$:

Tabla 2.1. Definición de los operadores Alfa y Beta

$\alpha : A \times A \rightarrow B$		
x	y	$\alpha\,(x, y)$
0	0	1
0	1	0
1	0	2
1	1	1

$\beta : B \times A \rightarrow A$		
x	y	$\beta\,(x, y)$
0	0	0
0	1	0
1	0	0
1	1	1
2	0	1
2	1	1

La operación binaria α exhibe algunas propiedades algebraicas, expuestas en la Tabla 2.2.

Tabla 2.2. Propiedades de la operación binaria α

isoargumentos en α	$\alpha(x, x) = 1$
intercambio de argumentos en α	$(x \leq y) \leftrightarrow \alpha(x, y) \leq \alpha(y, x)$
α creciente por la izquierda	$(x \leq y) \leftrightarrow [\alpha(x, z) \leq \alpha(y, z)]$
α decreciente por la derecha	$(x \leq y) \leftrightarrow [\alpha(z, x) \geq \alpha(z, y)]$
α distributiva por la derecha respecto a $\bigvee$	$\alpha\,[(x \bigvee y)\,, z] = \alpha(x, z) \bigvee \alpha(y, z)$
α distributiva por la derecha respecto a $\bigwedge$	$\alpha\,[(x \bigwedge y)\,, z] = \alpha(x, z) \bigwedge \alpha(y, z)$

Asimismo, en la Tabla 2.3 se muestran algunas propiedades de la operación binaria β.

Tabla 2.3. Propiedades de la operación binaria β

propiedad del 1	$\beta(1, x) = x$
isoargumentos en β	$\beta(x, x) = x \quad \forall x \in A$
β creciente por la izquierda	$(x \leq y) \rightarrow [\beta(x, z) \leq \beta(y, z)]$
β creciente por la derecha	$(x \leq y) \rightarrow [\beta(z, x) \leq \beta(z, y)]$
β distributiva por la derecha respecto a $\bigvee$	$\beta\,[(x \bigvee y)\,, z] = \beta\,(x, z) \bigvee \beta\,(y, z)$
β distributiva por la derecha respecto a $\bigwedge$	$\beta\,[(x \bigwedge y)\,, z] = \beta\,(x, z) \bigwedge \beta\,(y, z)$
β distributiva por la izquierda respecto a $\bigvee$	$\beta\,[x, (y \bigvee z)] = \beta\,(x, y) \bigvee \beta\,(x, z)$
β distributiva por la izquierda respecto a $\bigwedge$	$\beta\,[x, (y \bigwedge z)] = \beta\,(x, y) \bigwedge \beta\,(x, z)$

Por otro lado, en la Tabla 2.4 se presentan las propiedades de la aplicación combinada de ambas operaciones α y β.

Tabla 2.4. Propiedades de la aplicación combinada de las operaciones α y β

β es la inversa de α por la derecha	$\beta\,[\alpha(x, y), y] = x$
β es la inversa de α por la izquierda	$\beta\,[\alpha(x, y), x] = x$
isoargumentos en α como argumento de β	$\beta\,[\alpha(x, x), y] = y$

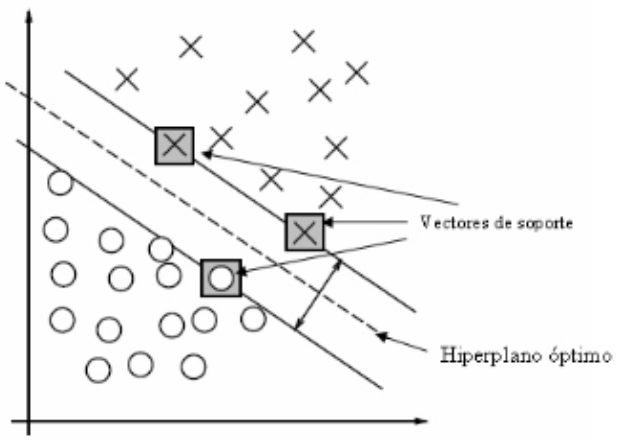


Figura 2.1: Support Vector Machine. Los vectores de soporte están indicados con un recuadro mientras que el hiperplano óptimo aparece como una línea punteada

2.3. Support Vector Machines

El algoritmo utilizado por las SVMs funciona como un clasificador biclase que minimiza simultáneamente el error empírico de clasificación y maximiza algunas características de las métricas involucradas [89], [90]. Como tal, este algoritmo está fuertemente basado en la teoría de aprendizaje estadístico desarrollada por Vapnik, Chervonenkis y otros, que dio lugar a la implementación de las SVMs durante la década de los noventas, en los Bell Laboratories de AT&T por Vapnik y sus colaboradores [74], [75].

El problema básico que ataca este modelo es el de de separar un hiperplano n-dimensional en dos clases, por medio de un hiperplano $n-1$-dimensional. Sin embargo, existen más de un hiperplano que logra este cometido. El objetivo de las SVMs es encontrar el hiperplano óptimo que mejor generalice la clasificación. Para ello, se utiliza el concepto de *vector de soporte*, que se refiere a los patrones más cercanos al hiperplano buscado; dichos patrones se encuentran en la frontera de las clases. Para encontrar ese hiperplano óptimo, se maximiza la distancia a los vectores de soporte. De manera informal, podemos afirmar que los vectores de soporte son los patrones que proporcionan más información para la tarea de clasificación [3]. Ahora bien, puede darse el caso de que existan patrones, cercanos a la frontera de las clases, que se comportan más como excepciones. Si dichos patrones se tomaran como vectores de soporte, degradarían la generalización de la clasificación. Para solucionar este problema, se incluye cierto margen de error, con lo cual se admite que ciertos patrones cercanos a la frontera no sean tomados como vectores de soporte, siempre y cuando se mantengan a una distancia menor o igual al margen de error establecido. Entonces, el problema de encontrar el hiperplano que divide las dos clases de manera óptima, se resuelve maximizando la distancia entre dicho hiperplano y los patrones más cercanos a la frontera de las clases (vectores de soporte), a la vez que se minimiza el error [75], [91], [92]. Un ejemplo de SVM se puede apreciar en la figura 2.1.

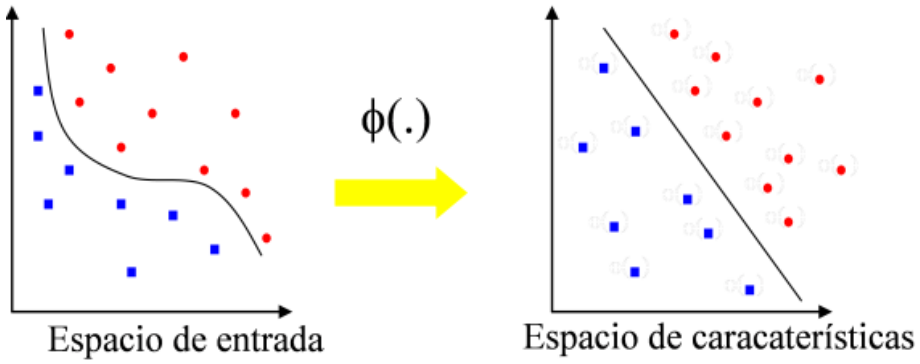


Figura 2.2: Ejemplo de función *kernel*

En caso de que el problema no sea linealmente separable en el espacio original, se utiliza una transformación por medio de una función *kernel*, para llevar los patrones a un espacio en una dimensión mayor, en el cual el problema sea linealmente separable [92]. La figura 2.2 muestra un ejemplo de mapeo de un espacio de entrada de dos dimensiones a un espacio de características de dos dimensiones, donde los datos no pueden ser separados por una función lineal en el espacio de entrada, pero sí pueden serlo en el espacio de características, donde la separación lineal es mucho más fácil.

Capítulo 3

Modelo Propuesto

Este capítulo es el más relevate del trabajo de tesis. Para la creación, diseño y fundamentación del nuevo modelo de máquinas asociativas Alfa-Beta con soporte vectorial, se utilizará el modelo de las memorias asociativas Alfa-Beta como punto de partida, tomando de éste los operadores α y β y sus propiedades; además, del modelo SVM (*Support Vector Machines*) se tomará la idea del vector de soporte, aunque el concepto del nuevo vector de soporte es totalmente diferente del concepto que usan las SVM. Es preciso hacer notar que parte de la estrategia para la creación del nuevo modelo, es no usar ni la teoría de los modelos asociativos Alfa-Beta ni la teoría de las SVM, sino crear un nuevo concepto teórico de vector de soporte, caracterizado con base en los operadores α y β y sus propiedades. Podría pensarse en el nuevo modelo como un tipo de SVM binarias.

Dado que el modelo de las máquinas asociativas Alfa-Beta con soporte vectorial es autoasociativo, se asume que se tiene un problema de reconocimiento de patrones, donde el conjunto fundamental es de la forma $\{(\mathbf{x}^\mu, \mathbf{x}^\mu) \mid \mu = 1, 2, ..., p\}$, con $\mathbf{x}^\mu \in A^n$ $\forall \mu \in \{1, 2, ..., p\}$, siendo $n, p \in \mathbb{Z}^+$ y $A = \{0, 1\}$.

Los vectores $\mathbf{x}^\mu$ son vectores columna, y el índice de las componentes es creciente de arriba hacia abajo, según se ilustra a continuación:

$$\mathbf{x}^\mu = \begin{pmatrix} x_1^\mu \\ x_2^\mu \\ \vdots \\ x_n^\mu \end{pmatrix} \in A^n$$

El capítulo consta de 6 secciones. En la sección 3.1, con el auxilio de un ejemplo gráfico, se realiza una descripción sencilla de los propósitos, las caracterísitcas, los alcances y las limitaciones de las máquinas asociativas Alfa-Beta con soporte vectorial; con esta descripción es posible notar las enormes diferencias que hay entre el nuevo modelo y las SVM clásicas, descritas en la sección 2.2.

La sección 3.2 contiene las definiciones y resultados preliminares que servirán de apoyo en el planteamiento formal del nuevo modelo de máquinas asociativas Alfa-Beta con soporte vectorial, cuyo concepto fundamental, el vector de soporte para el nuevo modelo, se presenta en la sección 3.3.

Además del vector de soporte como concepro fundamental, en la sección 3.4 de

este trabajo de tesis se introducen y ejemplifican dos transformadas originales: la Transformada $\boldsymbol{\tau}$ y la Transformada θ; estas dos nuevas transformadas son cruciales en la generación del algoritmo principal del nuevo modelo de máquinas asociativas Alfa-Beta con soporte vectorial.

Finalmente, en las secciones 3.5 y 3.6 se presentan, respectivamente, el algoritmo principal y el análisis de su complejidad.

3.1. Descripción de las máquinas asociativas Alfa-Beta con soporte vectorial - ejemplo gráfico

Una de las ideas originales del modelo propuesto es la de tratar de aprovechar aquella información que se repite en los patrones fundamentales. Así, primero se obtiene un patrón que contiene la información repetida en todos los patrones fundamentales; posteriormente, esta información es *eliminada* de los patrones fundamentales, quedando únicamente con la información que diferencia a un patrón fundamental de todos los demás. Esta información repetida es la que ha inspirado el nuevo concepto de vector de soporte.

Cuando se presenta al modelo ya creado un patrón desconocido (puede pertenecer al conjunto fundamental o no), mismo que se pretende reconocer con este modelo, se procederá a *eliminar* la información repetida, tomando como base el patrón que contiene esta información repetida, que no es otra cosa que el vector de soporte para ese conjunto fundamental.

Ahora, es necesario determinar, de alguna manera, cuál de los patrones fundamentales es menos diferente con respecto a este patrón desconocido, al que se le quitó la información repetida (vector de soporte). Se espera que el patrón fundamental menos diferente sea el más parecido. Sólo faltará volver a colocar la información que se eliminó en el patrón fundamental recuperado, y se tiene listo el patrón de salida.

Para ilustrar lo anterior, veamos un ejemplo.

Ejemplo 3.1 *Sea un conjunto fundamental como el mostrado en la figura 3.1. Llevar a cabo las fases de aprendizaje y recuperación del modelo propuesto.*

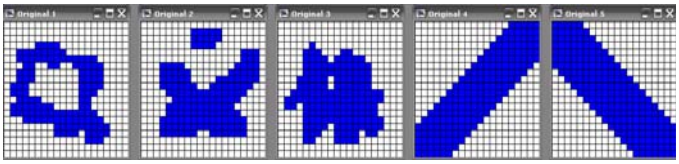


Figura 3.1: Conjunto fundamental

A partir de los patrones que conforman el conjunto fundamental, se obtiene un nuevo patrón que contiene la información presente en todos los patrones fundamentales: el vector de soporte. Esto se ilustra en la figura 3.2.

Este patrón se utiliza para *eliminar* la información presente en todos los patrones fundamentales. El resultado de lo anterior se puede ver en la figura 3.3.

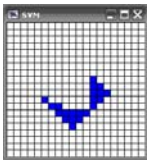


Figura 3.2: Patrón con la información repetida (vector de soporte)

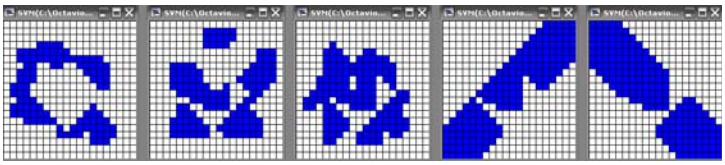


Figura 3.3: Conjunto fundamental con la información presente en todos los patrones fundamentales *eliminada*

Por otra lado, a partir de los patrones del conjunto fundamental original, se obtiene otro patrón con la información ausente en todos los patrones fundamentales. Esto se logra al negar los patrones fundamentales (puesto que los patrones son binarios; ver figura 3.4) y obtener el patrón con la información repetida en ellos (ver figura 3.5).

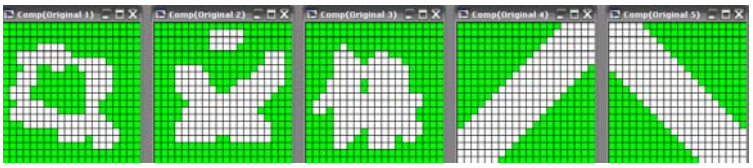


Figura 3.4: Conjunto fundamental negado

El resultado de restarle la información ausente a los patrones fundamentales se puede observar en la figura 3.6.

Ahora bien, al presentar un patrón al modelo, durante la fase de recuperación, como puede ser el mostrado en la figura 3.7, primero se elimina la información presente repetida (usando el patrón que se muestra en la figura 3.2). El resultado de eliminar dicha información se puede ver en la figura 3.8.

Por otro lado, se hace este mismo proceso anterior, pero con la imagen negada, para así quitarle al patrón la infromación ausente que se repite en los patrones fundamentales, usando como base el patrón mostrado en la figura 3.5. Entonces, se obtiene el patrón negado (ver figura 3.9) y se le *elimina* la información ausente repetida; el

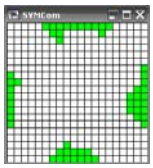


Figura 3.5: Patrón con la información repetida, ausente en todos los patrones fundamentales

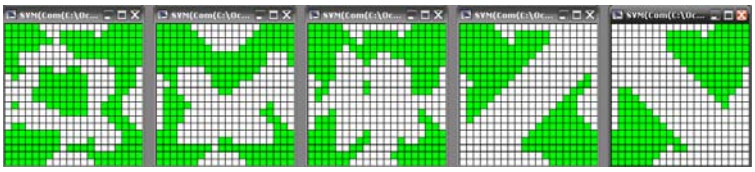


Figura 3.6: Conjunto fundamental con la información ausente en todos los patrones fundamentales *eliminada*

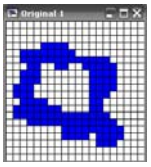


Figura 3.7: Patrón desconocido

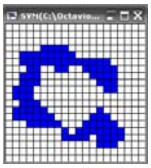


Figura 3.8: Patrón desconocido al que se ha *eliminado* la información presente repetida

resultado es la figura 3.10.

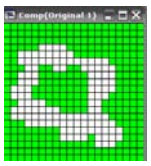


Figura 3.9: Patrón desconocido negado

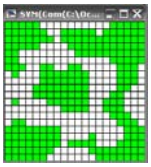


Figura 3.10: Patrón desconocido al que se ha *eliminado* la información ausente repetida

Al comparar el patrón desconocido al que se ha eliminado la información presente repetida con los patrones del conjunto fundamental sin la información presente repetida, y el patrón desconocido negado al que se ha eliminado la información ausente repetida con los patrones fundamentales sin la información ausente repetida, respectivamente, podemos ver que el patrón fundamental que menos diferencias tiene es el mostrado en la figura 3.11 para el primer caso, y el mostrado en la figura 3.12 para el segundo.

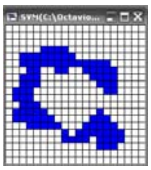


Figura 3.11: Patrón fundamental sin la información presente repetida menos diferente al patrón desconocido al que se ha *eliminado* la información presente repetida

Como en este caso, ambos patrones sin información repetida corresponden al mismo patrón fundamental, basta con incorporarles de nuevo la información eliminada para obtener el patrón de salida correspondiente al patrón desconocido presentado a la entrada, mismo que se muestra en la figura 3.13.

Este mismo proceso se lleva a cabo para los restantes patrones fundamentales. El patrón de entrada, su negado, el patrón de entrada sin la información presente repetida, el patrón negado sin la información ausente repetida, y el patrón recuperado, se muestran en las figuras 3.14, 3.15 y 3.16. En dichas figuras, el patrón de entrada se muestra en azul; seguido del patrón negado en verde; a continuación, estos dos

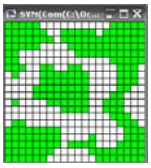


Figura 3.12: Patrón fundamental sin la información ausente repetida menos diferente al patrón desconocido al que se ha *eliminado* la información ausente repetida

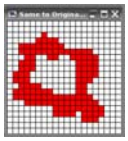


Figura 3.13: Patrón de salida

patrones sin la información repetida (la presente y la ausente, respectivamente), en azul y verde cada uno; y por último, el patrón recuperado en rojo.

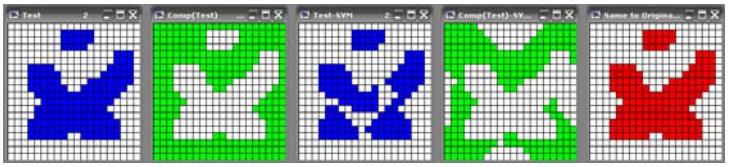


Figura 3.14: Recuperación del segundo patrón fundamental

Como se puede ver, el modelo recuperó correctamente cada uno de estos patrones fundamentales.

De este ejemplo es posible concluir para qué tipo de patrones se presta aplicar el modelo propuesto. Dado que la idea fundamental del modelo es aprovechar la información repetida en los patrones fundamentales, es de esperarse que se presente un mejor desempeño mientras más tengan en común los patrones con los que se trabaje. Así, las imágenes binarias son un tipo de patrón facilmente aplicable a este modelo. En parti- cular, imágenes binarias de caracteres escritos, ya sea digitalizadas de caracteres ecritos a mano en papel (como los patrones de la base de datos MNIST), o digitalizadas por un dispositivo especializado (como sería el dispositivo de entrada de un *handheld*). Sin embargo, cabe mencionar que este modelo no sólo funciona para imágenes binarias, sino para imágenes en otros planos de color, como puede ser escala de grises o RGB. La idea es que los patrones a tratar compartan áreas extensas con información común. Incluso pueden trabajarse patrones que no sean imágenes, siempre y cuando dichos patrones compartan información entre sí.

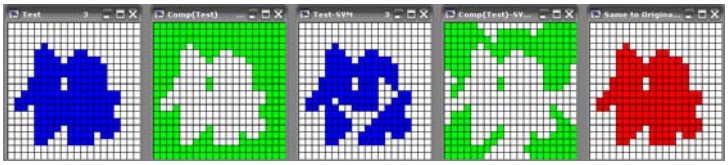


Figura 3.15: Recuperación del tercer patrón fundamental

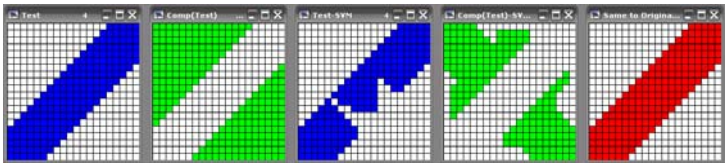


Figura 3.16: Recuperación del cuarto patrón fundamental

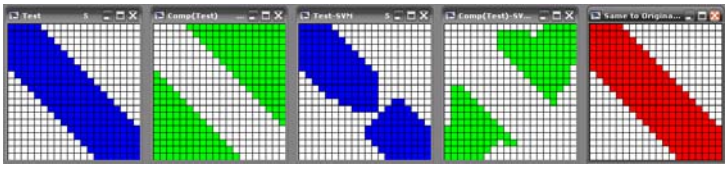


Figura 3.17: Recuperación del quinto patrón fundamental

3.2. Definiciones y resultados preliminares

En esta sección se plantean las definiciones y resultados preliminares, que son el apoyo para el planteamiento formal del nuevo modelo de máquinas asociativas Alfa-Beta con soporte vectorial.

Definición 3.2 *Un vector cero se define como el vector cuyas componentes son todas de valor* 0*, y se denota como* $\mathbf{0}$*.[101]*

La dimensión del vector $\mathbf{0}$ quedará determinada según el contexto donde se use. Así, el mismo símbolo $\mathbf{0}$ representa vectores cuyas componentes son todas de valor 0, no obstante que sus dimensiones sean diferentes.

Ejemplo 3.3 *Sea* $A = \{0, 1\}$*, y sean los vectores* $\mathbf{x}^1 = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$, $\mathbf{x}^2 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$, $\mathbf{x}^3 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$.

Como se puede ver, $\mathbf{x}^1 \in A^3$*,* $\mathbf{x}^2 \in A^5$*,* $\mathbf{x}^3 \in A^8$*; sin embargo, dado que para cada uno se cumple que* $x_i^\mu = 0\ \forall i$*, con* $\mu = 1, 2, 3$*; es claro que* $\mathbf{x}^1 = \mathbf{0}$ *en un contexto donde los vectores que se usan son de dimensión 3,* $\mathbf{x}^2 = \mathbf{0}$ *para dimensión 5 y* $\mathbf{x}^3 = \mathbf{0}$ *para dimensión 8.*

Definición 3.4 *Un vector uno se define como el vector cuyas componentes son todas de valor* 1*, y se denota como* $\mathbf{1}$*.*

La dimensión del vector $\mathbf{1}$ quedará determinada según el contexto donde se use. Así, el mismo símbolo $\mathbf{1}$ representa vectores cuyas componentes son todas de valor 1, no obstante que sus dimensiones sean diferentes.

Ejemplo 3.5 *Sea* $A = \{0, 1\}$*, y sean los vectores* $\mathbf{x}^1 = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$, $\mathbf{x}^2 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$, $\mathbf{x}^3 = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$.

Como se puede ver, $\mathbf{x}^1 \in A^4$*,* $\mathbf{x}^2 \in A^2$*,* $\mathbf{x}^3 \in A^9$*; sin embargo, dado que para cada uno se cumple que* $x_i^\mu = 1\ \forall i$*, con* $\mu = 1, 2, 3$*; es claro que* $\mathbf{x}^1 = \mathbf{1}$ *en un contexto donde los vectores que se usan son de dimensión 4,* $\mathbf{x}^2 = \mathbf{1}$ *para vectores de dimensión 2 y* $\mathbf{x}^3 = \mathbf{1}$ *para dimensión 9.*

Definición 3.6 *Sea $A = \{0, 1\}$, y sea $\mathbf{x}$ un vector columna $\mathbf{x} \in A^n$, con $n \in \mathbb{Z}^+$. Se define el vector negado de $\mathbf{x}$ como el vector denotado por $\overline{\mathbf{x}}$ tal que, para cada íncide $i \in \{1, 2, 3, ..., n\}$, las componentes de ambos vectores cumplen la expresión $\overline{x}_i = \neg x_i$, donde $\neg$ es la operación de negación booleana.*

Ejemplo 3.7 *Sea* $\mathbf{x} = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}$*; obtener* $\overline{\mathbf{x}}$*.*

De acuerdo con la Definición 3.6, se tiene que: $\overline{x}_1 = \neg x_1 = \neg 0 = 1$, $\overline{x}_2 = \neg x_2 = \neg 1 = 0$ y $\overline{x}_3 = \neg x_3 = \neg 0 = 1$; por ello, $\overline{\mathbf{x}} = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}$.

Definición 3.8 *Sea $A = \{0, 1\}$, y sea $\mathbf{x}$ un vector columna $\mathbf{x} \in A^n$, con $n \in \mathbb{Z}^+$; sea, además, $i \in \mathbb{Z}^+$ tal que $1 \leq i \leq n$. Se define el escalar $\sigma_i(\mathbf{x})$ como sigue:*

$$\sigma_i(\mathbf{x}) = \sum_{j=1}^{i} \beta\left(x_j, x_j\right) \text{ para } i \in \{1, 2, \ldots, n\}$$

Nota 3.9 *Obsérvese que la operación $\beta(x_j, x_j)$ es una forma alternativa de expresar la operación lógica AND del valor binario x_j consigo mismo; esto significa que el escalar $\sigma_i(\mathbf{x})$ representa el número de componentes con valor 1 que contiene el vector $\mathbf{x}$ en las primeras i componentes.*

Ejemplo 3.10 *Sea* $\mathbf{x} = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}$*; obtener $\sigma_i(\mathbf{x})$, para cada $i \in \{1, 2, \ldots, n\}$.*

Es claro que $\mathbf{x} \in A^n$, con $n = 3$. Por la definición 3.8, se pueden calcular 3 diferentes valores para $\sigma_i(\mathbf{x})$:

$$\begin{array}{ll} \sigma_1(\mathbf{x}) & = \beta(x_1, x_1) = \beta(0, 0) = 0 \\ \sigma_2(\mathbf{x}) & = \beta(x_1, x_1) + \beta(x_2, x_2) = \beta(0, 0) + \beta(1, 1) = 0 + 1 = 1 \\ \sigma_3(\mathbf{x}) & = \beta(x_1, x_1) + \beta(x_2, x_2) + \beta(x_3, x_3) = \beta(0, 0) + \beta(1, 1) + \beta(0, 0) \\ & = 0 + 1 + 0 = 1 \end{array}$$

Ejemplo 3.11 *Sean* $\mathbf{x} = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}$ *y* $\mathbf{y} = \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix}$*; obtener $\sigma_i(\mathbf{x})$, $\sigma_i(\mathbf{y})$, para cada $i \in \{1, 2, \ldots, n\}$.*

Es claro que $\mathbf{x} \in A^n$, $\mathbf{y} \in A^n$, ambos con $n = 3$. Por la definición 3.8, se pueden calcular 3 diferentes valores para $\sigma_i(\mathbf{x})$ y otros 3 para $\sigma_i(\mathbf{y})$:

$$\begin{aligned}
\sigma_1(\mathbf{x}) &= \beta(x_1, x_1) = \beta(1,1) = 1 \\
\sigma_2(\mathbf{x}) &= \beta(x_1, x_1) + \beta(x_2, x_2) = \beta(1,1) + \beta(1,1) = 1 + 1 = 2 \\
\sigma_3(\mathbf{x}) &= \beta(x_1, x_1) + \beta(x_2, x_2) + \beta(x_3, x_3) = \beta(1,1) + \beta(1,1) + \beta(0,0) \\
&= 1 + 1 + 0 = 2
\end{aligned}$$

$$\begin{aligned}
\sigma_1(\mathbf{y}) &= \beta(y_1, y_1) = \beta(0,0) = 0 \\
\sigma_2(\mathbf{y}) &= \beta(y_1, y_1) + \beta(y_2, y_2) = \beta(0,0) + \beta(1,1) = 0 + 1 = 1 \\
\sigma_3(\mathbf{y}) &= \beta(y_1, y_1) + \beta(y_2, y_2) + \beta(y_3, y_3) = \beta(0,0) + \beta(1,1) + \beta(1,1) \\
&= 0 + 1 + 1 = 2
\end{aligned}$$

Ejemplo 3.12 *Sea* $\mathbf{x}$ *un vector columna* $\mathbf{x} \in A^n$*, donde* $n = 8$ *y* $\mathbf{x} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \end{pmatrix}$*; obtener los diferentes valores de* $\sigma_i(\mathbf{x})$*, para cada* $i \in \{1, 2, \ldots, n\}$*.*

Por la definición 3.8, se pueden calcular 8 diferentes valores para $\sigma_i(\mathbf{x})$:

$$\begin{aligned}
\sigma_1(\mathbf{x}) &= 1 \\
\sigma_2(\mathbf{x}) &= 1 \\
\sigma_3(\mathbf{x}) &= 1 \\
\sigma_4(\mathbf{x}) &= 2 \\
\sigma_5(\mathbf{x}) &= 3 \\
\sigma_6(\mathbf{x}) &= 4 \\
\sigma_7(\mathbf{x}) &= 4 \\
\sigma_8(\mathbf{x}) &= 4
\end{aligned}$$

Ejemplo 3.13 *Sean* $\mathbf{y} = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}$ *y* $\mathbf{z} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}$*; obtener los diferentes valores de* $\sigma_i(\mathbf{y})$*,* $\sigma_i(\mathbf{z})$*, para cada* $i \in \{1, 2, \ldots, n\}$*.*

Es claro que $\mathbf{y}, \mathbf{z} \in A^n$ donde $n = 6$, por lo que los valores para $\sigma_i(\mathbf{y})$ y $\sigma_i(\mathbf{z})$ son los siguientes:

$$\begin{array}{ll} \sigma_1(\mathbf{y}) = 1 & \sigma_1(\mathbf{z}) = 0 \\ \sigma_2(\mathbf{y}) = 2 & \sigma_2(\mathbf{z}) = 0 \\ \sigma_3(\mathbf{y}) = 2 & \sigma_3(\mathbf{z}) = 1 \\ \sigma_4(\mathbf{y}) = 2 & \sigma_4(\mathbf{z}) = 1 \\ \sigma_5(\mathbf{y}) = 3 & \sigma_5(\mathbf{z}) = 1 \\ \sigma_6(\mathbf{y}) = 3 & \sigma_6(\mathbf{z}) = 2 \end{array}$$

Para esta operación, existen dos casos particulares, que son cuando se trabaja con vectores cuyas componentes tienen, o todas valores de 0, o todas valores de 1; esto es, los vectores $\mathbf{0}$ y $\mathbf{1}$. Lo anterior se puede apreciar en el siguiente ejemplo.

Ejemplo 3.14 *Sean* $\mathbf{0} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$ *y* $\mathbf{1} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$*; es claro que* $\mathbf{0}, \mathbf{1} \in A^n$ *donde* $n = 8$*, por lo que los valores para* $\sigma_i(\mathbf{0})$ *y* $\sigma_i(\mathbf{1})$ *son los siguientes:*

$$\begin{array}{ll} \sigma_1(\mathbf{0}) = 0 & \sigma_1(\mathbf{1}) = 1 \\ \sigma_2(\mathbf{0}) = 0 & \sigma_2(\mathbf{1}) = 2 \\ \sigma_3(\mathbf{0}) = 0 & \sigma_3(\mathbf{1}) = 3 \\ \sigma_4(\mathbf{0}) = 0 & \sigma_4(\mathbf{1}) = 4 \\ \sigma_5(\mathbf{0}) = 0 & \sigma_5(\mathbf{1}) = 5 \\ \sigma_6(\mathbf{0}) = 0 & \sigma_6(\mathbf{1}) = 6 \\ \sigma_7(\mathbf{0}) = 0 & \sigma_7(\mathbf{1}) = 7 \\ \sigma_8(\mathbf{0}) = 0 & \sigma_8(\mathbf{1}) = 8 \end{array}$$

Nota 3.15 *Como se puede apreciar* $\sigma_i(\mathbf{0}) = 0\ \forall i \in \{1, 2, \ldots, n\}$*. Esto se debe a que todas la componentes de* $\mathbf{0}$ *son* 0*, por lo que no existen componentes con valor de* 1*. Por otro lado,* $\sigma_i(\mathbf{1}) = i\ \forall i \in \{1, 2, \ldots, n\}$*. Lo anterior no tiene nada de raro, puesto que todas las componentes de* $\mathbf{1}$ *son* 1*, por lo que habrá tantos valores* 1 *como componentes se revisen; esto es,* i *componentes.*

Proposición 3.16 *Sea* $A = \{0, 1\}$*. Si* $\mathbf{x}$ *es un vector columna* $\mathbf{x} \in A^n$*, con* $n \in \mathbb{Z}^+$*; el escalar* $\sigma_n(\mathbf{x})$ *representa el número total de componentes con valor* 1 *que contiene el vector* $\mathbf{x}$*.*

Demostración.- Sea $j \in \mathbb{Z}^+$ tal que $1 \leq j \leq n$. Dado que $\mathbf{x} \in A^n$, x_j posee uno de dos valores posibles: $x_j = 0$ o $x_j = 1$; si $x_j = 0$ se tiene que $\beta(x_j, x_j) = 0$, y en el caso de que $x_j = 1$, $\beta(x_j, x_j) = 1$. Ahora, para calcular el valor $\sigma_n(\mathbf{x})$, se debe realizar la suma de todos los valores $\beta(x_j, x_j)$ con $1 \leq j \leq n$; cada vez que $x_j = 0$ el resultado de la suma no varía, pero cada vez que $x_j = 1$, el resultado de al suma se incrementa en 1, quedando, al llegar a $j = n$, el número total de componentes con valor 1: y es precisamente el valor de $\sigma_n(\mathbf{x})$.■

Nota 3.17 *En todo el capítulo, se usará el símbolo* ■ *para señalar el final de una demostración.*

Ejemplo 3.18 *Sean* $\mathbf{x} = \begin{pmatrix}1\\0\\1\\1\end{pmatrix}$, $\mathbf{y} = \begin{pmatrix}0\\0\\1\\0\end{pmatrix}$ *y* $\mathbf{z} = \begin{pmatrix}0\\1\\1\\0\end{pmatrix}$*; obtener* $\sigma_n(\mathbf{x})$, $\sigma_n(\mathbf{y})$ *y* $\sigma_n(\mathbf{z})$.

Es claro que $\mathbf{x}, \mathbf{y}, \mathbf{z} \in A^n$ donde $n = 4$. Entonces, por la definición :3.8 $\sigma_4(\mathbf{x}) = 3$, $\sigma_4(\mathbf{y}) = 1$, $\sigma_4(\mathbf{z}) = 2$.

Ejemplo 3.19 *Sean* $\mathbf{x}^1 = \begin{pmatrix}0\\0\\0\\1\\0\end{pmatrix}$ *y* $\mathbf{x}^2 = \begin{pmatrix}1\\1\\1\\0\\1\\0\\1\\0\end{pmatrix}$*; obtener* $\sigma_n(\mathbf{x}^1)$ *y* $\sigma_n(\mathbf{x}^2)$.

Como se puede ver $\mathbf{x}^1 \in A^5$, mientras que $\mathbf{x}^2 \in A^8$. Los valores para $\sigma_n(\mathbf{x}^1)$ y $\sigma_n(\mathbf{x}^2)$ son, respectivamente: $\sigma_5(\mathbf{x}^1) = 1$ y $\sigma_8(\mathbf{x}^2) = 5$.

Ejemplo 3.20 *Sean* $\mathbf{0} = \begin{pmatrix}0\\0\\0\\0\\0\end{pmatrix}$ *y* $\mathbf{1} = \begin{pmatrix}1\\1\\1\end{pmatrix}$*; obtener* $\sigma_n(\mathbf{0})$ *y* $\sigma_n(\mathbf{1})$.

Es claro que $\mathbf{0} \in A^5$, mientras que $\mathbf{1} \in A^3$. Los valores para $\sigma_n(\mathbf{0})$ y $\sigma_n(\mathbf{1})$ son, respectivamente: $\sigma_5(\mathbf{0}) = 0$ y $\sigma_3(\mathbf{1}) = 3$. Lo anterior concuerda con lo visto en el ejemplo 3.14.

Definición 3.21 *Sea* $A = \{0, 1\}$*, y sean* $\mathbf{x}$ *y* $\mathbf{z}$ *dos vectores columna* $\mathbf{x} \in A^n$ *y* $\mathbf{z} \in A^n$ *con* $\mathbf{x} \neq \mathbf{0}$*,* $\mathbf{x} \neq \mathbf{1}$*,* $\mathbf{z} \neq \mathbf{0}$*,* $\mathbf{z} \neq \mathbf{1}$ *y* $n \in \mathbb{Z}^+$*; además, sea* $c \in \mathbb{Z}^+$ *que cumple con la condición*

$$c = \bigwedge_{j=1}^{n} (j \text{ tal que } x_j = 1)$$

donde $\wedge$ *es el operador* mínimo*. Se define la Eliminación en* $\mathbf{z}$ *según* $\mathbf{x}$*, como la operación mediante la cual se realizan las siguientes dos acciones, en orden: se eliminan las componentes* x_c *y* z_c*; y además, los índices de las componentes* x_i *y* z_i *con* $c < i \leq n$*, si existen, se disminuyen en* 1*. Al finalizar estas dos acciones, se obtienen dos vectores transformados,* $\mathcal{E}_1(\mathbf{x})$ *y* $\mathcal{E}_1(\mathbf{z})$*. Si la Eliminación en* $\mathbf{z}$ *según* $\mathbf{x}$ *se aplica iteradamente, los vectores transformados en la* k*-ésima iteración, con* $k \in \mathbb{Z}^+$*, se denotan así:* $\mathcal{E}_k(\mathbf{x})$

y $\mathcal{E}_k(\mathbf{z})$*, siempre que la k-ésima iteración sea válida; es decir, que* $\mathcal{E}_{k-1}(\mathbf{x}) \neq \mathbf{0}$*; para ello, el cálculo de c se realiza así:*

$$c = \bigwedge_{j=1}^{n} (j \text{ tal que } [\mathcal{E}_{k-1}(\mathbf{x})]_j = 1)$$

Ejemplo 3.22 *Sean* $\mathbf{x} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$ *y* $\mathbf{z} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \end{pmatrix}$*; obtener la Eliminación en* $\mathbf{z}$ *según* $\mathbf{x}$*.*

Es claro que $\mathbf{x} \in A^n$ y $\mathbf{z} \in A^n$ con $n = 8$ para ambos vectores; ahora bien, el valor $c = 1$ cumple con la condición

$$c = \bigwedge_{j=1}^{8} (j \text{ tal que } x_j = 1)$$

puesto que $\{j \mid x_j = 1\} = \{1, 4, 6, 7\}$ y $\bigwedge_{j=1}^{8}(1, 4, 6, 7) = 1$. Entonces, de acuerdo con la definición 3.21, se realizan las dos siguientes acciones:

i) Dado que $c = 1$, se eliminan las componentes x_1 y z_1:

$$i) \quad \mathbf{x} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix} \text{ y } \mathbf{z} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \end{pmatrix} \longrightarrow \begin{pmatrix} x_2 = 0 \\ x_3 = 0 \\ x_4 = 1 \\ x_5 = 0 \\ x_6 = 1 \\ x_7 = 1 \\ x_8 = 0 \end{pmatrix} \text{ y } \begin{pmatrix} z_2 = 0 \\ z_3 = 1 \\ z_4 = 0 \\ z_5 = 1 \\ z_6 = 0 \\ z_7 = 1 \\ z_8 = 0 \end{pmatrix} \longrightarrow \begin{pmatrix} x_2 = 0 \\ x_3 = 0 \\ x_4 = 1 \\ x_5 = 0 \\ x_6 = 1 \\ x_7 = 1 \\ x_8 = 0 \end{pmatrix} \text{ y } \begin{pmatrix} z_2 = 0 \\ z_3 = 1 \\ z_4 = 0 \\ z_5 = 1 \\ z_6 = 0 \\ z_7 = 1 \\ z_8 = 0 \end{pmatrix}$$

ii) Se observa que sí existen las componentes x_i y z_i con $1 < i \leq n$, las cuales son $x_2, x_3, x_4, x_5, x_6, x_7, x_8, z_2, z_3, z_4, z_5, z_6, z_7$ y z_8; por ello, los índices de estas componentes se disminuyen en 1.

$$ii) \quad \begin{pmatrix} x_2 = 0 \\ x_3 = 0 \\ x_4 = 1 \\ x_5 = 0 \\ x_6 = 1 \\ x_7 = 1 \\ x_8 = 0 \end{pmatrix} \text{ y } \begin{pmatrix} z_2 = 0 \\ z_3 = 1 \\ z_4 = 0 \\ z_5 = 1 \\ z_6 = 0 \\ z_7 = 1 \\ z_8 = 0 \end{pmatrix} \longrightarrow \begin{pmatrix} x_1 = 0 \\ x_2 = 0 \\ x_3 = 1 \\ x_4 = 0 \\ x_5 = 1 \\ x_6 = 1 \\ x_7 = 0 \end{pmatrix} = \mathcal{E}_1(\mathbf{x}) \text{ y } \begin{pmatrix} z_1 = 0 \\ z_2 = 1 \\ z_3 = 0 \\ z_4 = 1 \\ z_5 = 0 \\ z_6 = 1 \\ z_7 = 0 \end{pmatrix} = \mathcal{E}_1(\mathbf{z})$$

Así, tenemos como resultado de la Eliminación en $\mathbf{z}$ según $\mathbf{x}$ que $\mathcal{E}_1(\mathbf{x}) \in \mathbf{A}^7$, $\mathcal{E}_1(\mathbf{z}) \in \mathbf{A}^7$, con $\mathcal{E}_1(\mathbf{x}) = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$ y $\mathcal{E}_1(\mathbf{z}) = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \end{pmatrix}$.

Ejemplo 3.23 *Sean* $\mathbf{x} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \end{pmatrix}$ *y* $\mathbf{z} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}$ *; obtener la Eliminación en* $\mathbf{z}$ *según* $\mathbf{x}$.

Es claro que $\mathbf{x} \in A^n$ y $\mathbf{z} \in A^n$ con $n = 7$ para ambos casos; ahora bien, $c = 2$ cumple con la condición

$$c = \bigwedge_{j=1}^{7} (j \text{ tal que } x_j = 1)$$

Entonces, de acuerdo con la definición 3.21, se realizan las dos siguientes acciones:
i) Dado que $c = 2$, se eliminan las componentes x_2 y z_2:

$$i) \quad \mathbf{x} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \end{pmatrix} \text{ y } \mathbf{z} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \longrightarrow \begin{pmatrix} x_1 = 0 \\ \\ x_3 = 0 \\ x_4 = 1 \\ x_5 = 1 \\ x_6 = 0 \\ x_7 = 1 \end{pmatrix} \text{ y } \begin{pmatrix} z_1 = 1 \\ \\ z_3 = 0 \\ z_4 = 0 \\ z_5 = 0 \\ z_6 = 1 \\ z_7 = 0 \end{pmatrix} \longrightarrow \begin{pmatrix} x_1 = 0 \\ x_3 = 0 \\ x_4 = 1 \\ x_5 = 1 \\ x_6 = 0 \\ x_7 = 1 \end{pmatrix} \text{ y } \begin{pmatrix} z_1 = 1 \\ z_3 = 0 \\ z_4 = 0 \\ z_5 = 0 \\ z_6 = 1 \\ z_7 = 0 \end{pmatrix}$$

ii) Se observa que sí existen las componentes x_i y z_i con $2 < i \leq n$, las cuales son $x_3, x_4, x_5, x_6, x_7, z_3, z_4, z_5, z_6$ y z_7; por ello, los índices de estas componentes se disminuyen en 1.

$$ii) \quad \begin{pmatrix} x_1 = 0 \\ x_3 = 0 \\ x_4 = 1 \\ x_5 = 1 \\ x_6 = 0 \\ x_7 = 1 \end{pmatrix} \text{ y } \begin{pmatrix} z_1 = 1 \\ z_3 = 0 \\ z_4 = 0 \\ z_5 = 0 \\ z_6 = 1 \\ z_7 = 0 \end{pmatrix} \longrightarrow \begin{pmatrix} x_1 = 0 \\ x_2 = 0 \\ x_3 = 1 \\ x_4 = 1 \\ x_5 = 0 \\ x_6 = 1 \end{pmatrix} = \mathcal{E}_1(\mathbf{x}) \text{ y } \begin{pmatrix} z_1 = 1 \\ z_2 = 0 \\ z_3 = 0 \\ z_4 = 0 \\ z_5 = 1 \\ z_6 = 0 \end{pmatrix} = \mathcal{E}_1(\mathbf{z})$$

Así, tenemos como resultado de la Eliminación en $\mathbf{z}$ según $\mathbf{x}$ que $\mathcal{E}_1(\mathbf{x}) \in \mathbf{A}^6$, $\mathcal{E}_1(\mathbf{z}) \in \mathbf{A}^6$, con $\mathcal{E}_1(\mathbf{x}) = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \end{pmatrix}$ y $\mathcal{E}_1(\mathbf{z}) = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}$.

Ejemplo 3.24 *Sean* $\mathbf{x} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$ *y* $\mathbf{z} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}$ *; obtener la Eliminación en* $\mathbf{z}$ *según* $\mathbf{x}$.

Es claro que $\mathbf{x} \in A^n$ y $\mathbf{z} \in A^n$ con $n = 9$ y $c = 4$, así que al realizar las dos acciones de la definición 3.21 se obtiene lo siguiente:

$$i) \quad \mathbf{x} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix} \text{ y } \mathbf{z} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} \longrightarrow \begin{pmatrix} x_1 = 0 \\ x_2 = 0 \\ x_3 = 0 \\ \\ x_5 = 1 \\ x_6 = 0 \\ x_7 = 1 \\ x_8 = 1 \\ x_9 = 0 \end{pmatrix} \text{ y } \begin{pmatrix} z_1 = 1 \\ z_2 = 0 \\ z_3 = 0 \\ \\ z_5 = 0 \\ z_6 = 1 \\ z_7 = 0 \\ z_8 = 0 \\ z_9 = 1 \end{pmatrix} \longrightarrow \begin{pmatrix} x_1 = 0 \\ x_2 = 0 \\ x_3 = 0 \\ x_5 = 1 \\ x_6 = 0 \\ x_7 = 1 \\ x_8 = 1 \\ x_9 = 0 \end{pmatrix} \text{ y } \begin{pmatrix} z_1 = 1 \\ z_2 = 0 \\ z_3 = 0 \\ z_5 = 0 \\ z_6 = 1 \\ z_7 = 0 \\ z_8 = 0 \\ z_9 = 1 \end{pmatrix}$$

$$ii) \quad \begin{pmatrix} x_1 = 0 \\ x_2 = 0 \\ x_3 = 0 \\ x_5 = 1 \\ x_6 = 0 \\ x_7 = 1 \\ x_8 = 1 \\ x_9 = 0 \end{pmatrix} \text{ y } \begin{pmatrix} z_1 = 1 \\ z_2 = 0 \\ z_3 = 0 \\ z_5 = 0 \\ z_6 = 1 \\ z_7 = 0 \\ z_8 = 0 \\ z_9 = 1 \end{pmatrix} \longrightarrow \begin{pmatrix} x_1 = 0 \\ x_2 = 0 \\ x_3 = 0 \\ x_4 = 1 \\ x_5 = 0 \\ x_6 = 1 \\ x_7 = 1 \\ x_8 = 0 \end{pmatrix} = \mathcal{E}_1(\mathbf{x}) \text{ y } \begin{pmatrix} z_1 = 1 \\ z_2 = 0 \\ z_3 = 0 \\ z_4 = 0 \\ z_5 = 1 \\ z_6 = 0 \\ z_7 = 0 \\ z_8 = 1 \end{pmatrix} = \mathcal{E}_1(\mathbf{z})$$

Así, el resultado de la Eliminación en $\mathbf{z}$ según $\mathbf{x}$ es $\mathcal{E}_1(\mathbf{x}) \in \mathbf{A}^8, \mathcal{E}_1(\mathbf{z}) \in \mathbf{A}^8, \mathcal{E}_1(\mathbf{x}) = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$

y $\mathcal{E}_1(\mathbf{z}) = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}$.

Proposición 3.25 *Sea $A = \{0, 1\}$. Si $\mathbf{x}$ y $\mathbf{z}$ son dos vectores columna $\mathbf{x} \in A^n$ y $\mathbf{z} \in A^n$ con $\mathbf{x} \neq \mathbf{0}$, $\mathbf{x} \neq \mathbf{1}$, $\mathbf{z} \neq \mathbf{0}$, $\mathbf{z} \neq \mathbf{1}$ y $n \in \mathbb{Z}^+$, se cumple que al aplicar una vez la operación Eliminación en $\mathbf{z}$ según $\mathbf{x}$, la dimensión de cada vector, tanto $\mathcal{E}_1(\mathbf{x})$ como $\mathcal{E}_1(\mathbf{z})$, es $n - 1$.*

Demostración.- Hay dos casos mutuamente exclusivos.

CASO 1 El valor de c en la definición 3.21 es n. Esto significa que el valor más pequeño de j tal que $x_j = 1$ es n; además, x_j es la única componente del vector $\mathbf{x}$ cuyo valor es 1, y por ello es el único valor de índice para el que se elimina una componente tanto en el vector $\mathbf{x} \in A^n$ como en el vector $\mathbf{z} \in A^n$, quedando ambos vectores con $n - 1$ componentes; es decir, $\mathcal{E}_1(\mathbf{x}) \in A^{n-1}$ y $\mathcal{E}_1(\mathbf{z}) \in A^{n-1}$. Entonces, la dimensión de cada vector, tanto $\mathcal{E}_1(\mathbf{x})$ como $\mathcal{E}_1(\mathbf{z})$, es $n - 1$.

CASO 2 El valor de c en la definición 3.21 es tal que $1 \leq c < n$. Esto significa que existen al menos una componente x_i y al menos una componente z_i con $c < i \leq n$; los índices de estas compnentes, según la definición 3.21, se disminuyen en 1 incluyendo, por supuesto, a las últimas componentes de ambos vectores, x_n y z_n, las cuales se convierten en x_{n-1} y z_{n-1}, quedando así ambos vectores con $n - 1$ componentes; es decir, $\mathcal{E}_1(\mathbf{x}) \in A^{n-1}$ y $\mathcal{E}_1(\mathbf{z}) \in A^{n-1}$. Entonces, la dimensión de cada vector, tanto $\mathcal{E}_1(\mathbf{x})$ como $\mathcal{E}_1(\mathbf{z})$, es $n - 1$.■

Ejemplo 3.26 *Sean $\mathbf{x} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$ y $\mathbf{z} = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 1 \end{pmatrix}$; obtener la Eliminación en $\mathbf{z}$ según $\mathbf{x}$.*

Para este caso $c = 4$; como $\mathbf{x}, \mathbf{z} \in A^4$ es claro que $n = 4$, así que $c = 4 = n$, por lo que nos encontramos en el primer caso de la proposición 3.25. Por la definición 3.21, tenemos que $\mathcal{E}_1(\mathbf{x}) = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$ y $\mathcal{E}_1(\mathbf{z}) = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}$, de donde resulta que $\mathcal{E}_1(\mathbf{x}) \in A^3$ y $\mathcal{E}_1(\mathbf{z}) \in A^3$, siendo entonces $n - 1$ la dimensión de $\mathcal{E}_1(\mathbf{x})$ y de $\mathcal{E}_1(\mathbf{z})$.

Ejemplo 3.27 *Sean* $\mathbf{x} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$ *y* $\mathbf{z} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{pmatrix}$ *; obtener la Eliminación en* $\mathbf{z}$ *según* $\mathbf{x}$*.*

Ahora tenemos que $c = 12$; como $\mathbf{x}, \mathbf{z} \in A^{12}$ es claro que $n = 12$, así que $c = 12 = n$, por lo que nos encontramos en el primer caso de la proposición 3.25. Por la definición 3.21, tenemos que $\mathcal{E}_1(\mathbf{x}) = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$ y $\mathcal{E}_1(\mathbf{z}) = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}$, de donde resulta que $\mathcal{E}_1(\mathbf{x}) \in A^{11}$ y $\mathcal{E}_1(\mathbf{z}) \in A^{11}$, siendo entonces $n - 1$ la dimensión de $\mathcal{E}_1(\mathbf{x})$ y de $\mathcal{E}_1(\mathbf{z})$.

Ejemplo 3.28 *Sean* $\mathbf{x} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \end{pmatrix}$ *y* $\mathbf{z} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$ *; obtener la Eliminación en* $\mathbf{z}$ *según* $\mathbf{x}$*.*

Para este caso $c = 3$; como $\mathbf{x}, \mathbf{z} \in A^8$ es claro que $n = 8$, así que $c = 3 \neq 8 = n$; más bien, $1 \leq c < 8$, por lo que nos encontramos en el segundo caso de la proposición 3.25. Por la definición 3.21, tenemos que $\mathcal{E}_1(\mathbf{x}) = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \end{pmatrix}$ y $\mathcal{E}_1(\mathbf{z}) = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$, de donde resulta que $\mathcal{E}_1(\mathbf{x}) \in A^7$ y $\mathcal{E}_1(\mathbf{z}) \in A^7$, siendo entonces $n - 1$ la dimensión de $\mathcal{E}_1(\mathbf{x})$ y de $\mathcal{E}_1(\mathbf{z})$.

Ejemplo 3.29 *Sean* $\mathbf{x} = \begin{pmatrix}1\\0\\0\\1\\0\\1\\1\end{pmatrix}$ *y* $\mathbf{z} = \begin{pmatrix}0\\0\\1\\0\\1\\0\\1\end{pmatrix}$*; obtener la Eliminación en* $\mathbf{z}$ *según* $\mathbf{x}$*.*

Para este caso $c = 1$; como $\mathbf{x}, \mathbf{z} \in A^7$ es claro que $n = 7$, así que $c = 1 \neq 7 = n$. más bien, $1 \leq c < 7$, por lo que nos encontramos en el segundo caso de la proposición 3.25. Por la definición 3.21, tenemos que $\mathcal{E}_1(\mathbf{x}) = \begin{pmatrix}0\\0\\1\\0\\1\\1\end{pmatrix}$ y $\mathcal{E}_1(\mathbf{z}) = \begin{pmatrix}0\\1\\0\\1\\0\\1\end{pmatrix}$, de donde resulta que $\mathcal{E}_1(\mathbf{x}) \in A^6$ y $\mathcal{E}_1(\mathbf{z}) \in A^6$, siendo entonces $n - 1$ la dimensión de $\mathcal{E}_1(\mathbf{x})$ y de $\mathcal{E}_1(\mathbf{z})$.

Definición 3.30 *Sea* $A = \{0, 1\}$*, y sean* $\mathbf{x}$ *y* $\mathbf{y}$ *dos vectores columna* $\mathbf{x} \in A^n$ *y* $\mathbf{y} \in A^n$ *con* $\mathbf{x} \neq \mathbf{0}$*,* $\mathbf{x} \neq \mathbf{1}$*,* $\mathbf{y} \neq \mathbf{0}$*,* $\mathbf{y} \neq \mathbf{1}$ *y* $n \in \mathbb{Z}^+$*. Se define la Restricción de* $\mathbf{y}$ *por* $\mathbf{x}$*, y se denota como* $\mathbf{y}|_{\mathbf{x}}$*, al vector que resulta, a partir del vector* $\mathbf{y}$*, de aplicar iteradamente la operación Eliminación en* $\mathbf{y}$ *según* $\mathbf{x}$*, para todas y cada una de las componentes de* $\mathbf{x}$ *con valor* 1*.*

Ejemplo 3.31 *Sean* $\mathbf{x} = \begin{pmatrix}1\\0\\0\\1\\0\\1\\1\\0\end{pmatrix}$ *y* $\mathbf{y} = \begin{pmatrix}0\\0\\1\\0\\1\\0\\1\\0\end{pmatrix}$*; obtener* $\mathbf{y}|_{\mathbf{x}}$*, que es la Restricción de* $\mathbf{y}$ *por* $\mathbf{x}$*.*

De acuerdo con la definición 3.30, para obtener $\mathbf{y}|_{\mathbf{x}}$ es necesario llevar a cabo la Eliminación en $\mathbf{y}$ según $\mathbf{x}$, para todas y cada una de las componentes de $\mathbf{x}$ con valor 1. Entonces, como se vio en el ejemplo 3.22, para $c = 1$:

$$\mathcal{E}_1(\mathbf{x}) = \begin{pmatrix}0\\0\\1\\0\\1\\1\\0\end{pmatrix} \text{ y } \mathcal{E}_1(\mathbf{y}) = \begin{pmatrix}0\\1\\0\\1\\0\\1\\0\end{pmatrix}$$

En la segunda iteración de la operación Eliminación en $\mathbf{y}$ según $\mathbf{x}$, al calcular el valor de c se obtiene $c = 3$:

$$\mathcal{E}_2(\mathbf{x}) = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix} \text{ y } \mathcal{E}_2(\mathbf{y}) = \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \end{pmatrix}$$

En la tercera iteración $c = 4$:

$$\mathcal{E}_3(\mathbf{x}) = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \text{ y } \mathcal{E}_3(\mathbf{y}) = \begin{pmatrix} 0 \\ 1 \\ 1 \\ 1 \\ 0 \end{pmatrix}$$

Para la cuarte iteración $c = 4$:

$$\mathcal{E}_4(\mathbf{x}) = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \text{ y } \mathcal{E}_4(\mathbf{y}) = \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$$

No es posible realizar la quinta iteración de la operación Eliminación en $\mathbf{y}$ según $\mathbf{x}$, dado que $\mathcal{E}_4(\mathbf{x}) = \mathbf{0}$.

Así, tenemos que la Eliminación en $\mathbf{y}$ según $\mathbf{x}$, denotada por $\mathbf{y}|_{\mathbf{x}} = \mathcal{E}_4(\mathbf{y}) = \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$.

Nótese que tanto la tercera como la cuarta Eliminaciones en $\mathbf{y}$ según $\mathbf{x}$ se hicieron para la componente $c = 4$. Esto no representa ningún problema, puesto que para la tercera Eliminación, c se calcula a partir de $\mathcal{E}_2(\mathbf{x})$ mientras que para la cuarta se utiliza $\mathcal{E}_3(\mathbf{x})$ para calcular c.

Ejemplo 3.32 *Sean* $\mathbf{x} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}$ *y* $\mathbf{y} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$ *; obtener* $\mathbf{y}|_{\mathbf{x}}$.

Para obtener $\mathbf{y}|_{\mathbf{x}}$ es necesario llevar a cabo la Eliminación en $\mathbf{y}$ según $\mathbf{x}$ tres veces, puesto que $\mathbf{x}$ tiene 3 componentes con valor de 1; así, tenemos para $c = 1$, $c = 5$ y

$c = 7$, respectivamente:

$$\mathcal{E}_1(\mathbf{x}) = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} \text{ y } \mathcal{E}_1(\mathbf{y}) = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$$

$$\mathcal{E}_2(\mathbf{x}) = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} \text{ y } \mathcal{E}_2(\mathbf{y}) = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 0 \end{pmatrix}$$

$$\mathcal{E}_3(\mathbf{x}) = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \text{ y } \mathcal{E}_3(\mathbf{y}) = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$$

por lo que $\mathbf{y}|_{\mathbf{x}} = \mathcal{E}_3(\mathbf{y}) = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$.

Ejemplo 3.33 *Sean* $\mathbf{x} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{pmatrix}$ *y* $\mathbf{y} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$ *; obtener* $\mathbf{y}|_{\mathbf{x}}$.

En este caso se realizan 6 Eliminaciones en $\mathbf{y}$ según $\mathbf{x}$:

$$\mathcal{E}_1(\mathbf{x}) = \begin{pmatrix}0\\0\\1\\0\\1\\1\\0\\0\\1\\0\\1\end{pmatrix} \text{ y } \mathcal{E}_1(\mathbf{y}) = \begin{pmatrix}0\\0\\0\\1\\0\\0\\0\\0\\0\\0\\1\end{pmatrix} \longrightarrow \mathcal{E}_2(\mathbf{x}) = \begin{pmatrix}0\\0\\0\\1\\1\\0\\0\\1\\0\\1\end{pmatrix} \text{ y } \mathcal{E}_2(\mathbf{y}) = \begin{pmatrix}0\\0\\1\\0\\0\\0\\0\\0\\0\\0\\1\end{pmatrix}$$

$$\longrightarrow \mathcal{E}_3(\mathbf{x}) = \begin{pmatrix}0\\0\\0\\1\\0\\0\\1\\0\\1\end{pmatrix} \text{ y } \mathcal{E}_3(\mathbf{y}) = \begin{pmatrix}0\\0\\1\\0\\0\\0\\0\\0\\0\\1\end{pmatrix} \longrightarrow \mathcal{E}_4(\mathbf{x}) = \begin{pmatrix}0\\0\\0\\0\\0\\1\\0\\1\end{pmatrix} \text{ y } \mathcal{E}_4(\mathbf{y}) = \begin{pmatrix}0\\0\\1\\0\\0\\0\\0\\0\\1\end{pmatrix}$$

$$\longrightarrow \mathcal{E}_5(\mathbf{x}) = \begin{pmatrix}0\\0\\0\\0\\0\\0\\0\\1\end{pmatrix} \text{ y } \mathcal{E}_5(\mathbf{y}) = \begin{pmatrix}0\\0\\1\\0\\0\\0\\0\\1\end{pmatrix} \longrightarrow \mathcal{E}_6(\mathbf{x}) = \begin{pmatrix}0\\0\\0\\0\\0\\0\\0\\0\end{pmatrix} \text{ y } \mathcal{E}_6(\mathbf{y}) = \begin{pmatrix}0\\0\\1\\0\\0\\0\\0\\0\end{pmatrix}$$

con lo que se obtiene $\mathbf{y}|_{\mathbf{x}} = \begin{pmatrix}0\\0\\1\\0\\0\\0\end{pmatrix}$.

Ejemplo 3.34 *Sean* $\mathbf{x}^1 = \begin{pmatrix}0\\0\\1\\0\\1\\0\\1\\0\end{pmatrix}$, $\mathbf{y}^1 = \begin{pmatrix}1\\0\\0\\1\\0\\1\\1\\0\end{pmatrix}$, $\mathbf{x}^2 = \begin{pmatrix}1\\0\\0\\1\\0\\1\\0\\1\\1\end{pmatrix}$ *y* $\mathbf{y}^2 = \begin{pmatrix}0\\0\\1\\0\\1\\0\\1\\0\\1\end{pmatrix}$*; obtener la*

Restricción de $\mathbf{y}^1$ *por* $\mathbf{x}^1$, $\mathbf{y}^1\big|_{\mathbf{x}^1}$*; la Restricción de* $\mathbf{y}^2$ *por* $\mathbf{x}^2$, $\mathbf{y}^2\big|_{\mathbf{x}^2}$*; la Restricción de* $\mathbf{x}^1$ *por* $\mathbf{y}^1$, $\mathbf{x}^1\big|_{\mathbf{y}^1}$ *y la Restricción de* $\mathbf{x}^2$ *por* $\mathbf{y}^2$, $\mathbf{x}^2\big|_{\mathbf{y}^2}$.

Al aplicar la definición 3.30, $\mathbf{y}^1\big|_{\mathbf{x}^1} = \mathcal{E}_3(\mathbf{y}^1) = \begin{pmatrix}1\\0\\1\\1\\0\end{pmatrix}$, $\mathbf{y}^2\big|_{\mathbf{x}^2} = \mathcal{E}_4(\mathbf{y}^2) = \begin{pmatrix}0\\1\\1\end{pmatrix}$,

$\mathbf{x}^1\big|_{\mathbf{y}^1} = \mathcal{E}_4(\mathbf{x}^1) = \begin{pmatrix}0\\1\\1\\0\end{pmatrix}$ y $\mathbf{x}^2\big|_{\mathbf{y}^2} = \mathcal{E}_3(\mathbf{x}^2) = \begin{pmatrix}1\\0\\1\\1\end{pmatrix}$.

Proposición 3.35 *Sea* $A = \{0,1\}$. Si $\mathbf{x}$ *y* $\mathbf{y}$ *son dos vectores columna* $\mathbf{x} \in A^n$ *y* $\mathbf{y} \in A^n$ *con* $\mathbf{x} \neq \mathbf{0}$, $\mathbf{x} \neq \mathbf{1}$, $\mathbf{y} \neq \mathbf{0}$, $\mathbf{y} \neq \mathbf{1}$ *y* $n \in \mathbb{Z}^+$, *se cumplen las siguientes dos afirmaciones:*

i) Para obtener $\mathbf{y}|_{\mathbf{x}}$, *se requiere aplicar* $\sigma_n(\mathbf{x})$ *veces la operación Eliminación en* $\mathbf{y}$ *según* $\mathbf{x}$.

ii) La dimensión del vector $\mathbf{y}|_{\mathbf{x}}$ *es* $n - \sigma_n(\mathbf{x})$; *es decir,* $\mathbf{y}|_{\mathbf{x}} \in A^{n-\sigma_n(\mathbf{x})}$.

Demostración.-

i) De acuerdo con la Definición 3.21, cada vez que se aplica la la operación Eliminación en $\mathbf{y}$ según $\mathbf{x}$ se elimina en el vector $\mathbf{x}$ una componente con valor 1; lo cual está acorde con la proposición 3.25, porque la dimensión del vector $\mathcal{E}_1(\mathbf{x})$ es $n-1$. Por otro lado, según la Definición 3.30, para obtener $\mathbf{y}|_{\mathbf{x}}$ se aplica iteradamente la operación Eliminación en $\mathbf{y}$ según $\mathbf{x}$, para todas y cada una de las componentes de $\mathbf{x}$ con valor 1; pero según la proposición 3.16 el escalar $\sigma_n(\mathbf{x})$ representa precisamentre el número total de componentes con valor 1 que contiene el vector $\mathbf{x}$. Por lo tanto, se concluye que para obtener $\mathbf{y}|_{\mathbf{x}}$, se requiere aplicar $\sigma_n(\mathbf{x})$ veces la operación Eliminación en $\mathbf{y}$ según $\mathbf{x}$.

ii) De acuerdo con la Definición 3.21, al aplicar una vez la operación Eliminación en $\mathbf{y}$ según $\mathbf{x}$, se obtienen los dos vectores transformados $\mathcal{E}_1(\mathbf{x})$ y $\mathcal{E}_1(\mathbf{y})$, ambos de dimensión $n-1$. En la segunda iteración, cada uno de los vectores transformados, $\mathcal{E}_2(\mathbf{x})$ y $\mathcal{E}_2(\mathbf{y})$, será de dimensión $(n-1)-1 = n-2$; así, al llegar a la última iteración, que es la iteración $\sigma_n(\mathbf{x})$ según la parte i) de esta proposición, se tendrá que el vector transformado $\mathcal{E}_{\sigma_n(\mathbf{x})}(\mathbf{y})$, el cual es precismante $\mathbf{y}|_{\mathbf{x}}$, tiene por dimensión el valor $n - \sigma_n(\mathbf{x})$; es decir, $\mathbf{y}|_{\mathbf{x}} \in A^{n-\sigma_n(\mathbf{x})}$.■

Definición 3.36 *Sea* $A = \{0,1\}$, *y sean* $\mathbf{x}$ *y* $\mathbf{z}$ *dos vectores columna* $\mathbf{x} \in A^n$ *y* $\mathbf{z} \in A^m$ *con* $\mathbf{x} \neq \mathbf{0}$, $\mathbf{x} \neq \mathbf{1}$, $\mathbf{z} \neq \mathbf{0}$, $\mathbf{z} \neq \mathbf{1}$, $n, m \in \mathbb{Z}^+$, *y* $m = n - \sigma_n(\mathbf{x})$; *además, sea* $c \in \mathbb{Z}^+$ *que cumple con la condición*

$$c = \bigwedge_{j=1}^{n} (j \text{ tal que } x_j = 1)$$

donde $\wedge$ *es el operador* mínimo. *Se define la Inserción en* $\mathbf{z}$ *según* $\mathbf{x}$, *como la operación mediante la cual se realizan las siguientes tres acciones, en orden: los índices de las*

componentes z_i con $c \leq i \leq m$, si existen, se aumentan en 1; se inserta una componente en la posición c del vector $\mathbf{z}$, de modo que $z_c = 1$; y se ejecuta la siguiente asignación: $x_c = 0$. Al finalizar estas tres acciones, se obtienen dos vectores transformados, $\mathcal{I}_1(\mathbf{x})$ y $\mathcal{I}_1(\mathbf{z})$. Si la Inserción en $\mathbf{z}$ según $\mathbf{x}$ se aplica iteradamente, los vectores transformados en la k-ésima iteración, con $k \in \mathbb{Z}^+$, se denotan así: $\mathcal{I}_k(\mathbf{x})$ y $\mathcal{I}_k(\mathbf{z})$, siempre que la k-ésima iteración sea válida; es decir, que $\mathcal{I}_{k-1}(\mathbf{x}) \neq \mathbf{0}$; y para ello, el cálculo de c se realiza así:

$$c = \bigwedge_{j=1}^{n} (j \text{ tal que } [\mathcal{I}_{k-1}(\mathbf{x})]_j = 1)$$

Nota 3.37 *De algún modo, la operación Inserción en $\mathbf{z}$ según $\mathbf{x}$ es una especie de inversa (bajo ciertas condiciones que se evidenciarán en un teorema posterior) de la operación Eliminación en $\mathbf{z}$ según $\mathbf{x}$*

Ejemplo 3.38 *Sean* $\mathbf{x} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}$ *y* $\mathbf{z} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \end{pmatrix}$*; obtener la Inserción en $\mathbf{z}$ según $\mathbf{x}$.*

Como se puede ver, $\mathbf{x} \in A^8$, $n = 8$, $\sigma_n(\mathbf{x}) = 3$, $n - \sigma_n(\mathbf{x}) = 8 - 3 = 5$ y $\mathbf{z} \in A^5$, por lo que $m = n - \sigma_n(\mathbf{x})$, valores acordes a la definición 3.36; el valor $c = 1$ cumple con la condición

$$c = \bigwedge_{j=1}^{n} (j \text{ tal que } x_j = 1)$$

Ahora, se llevan a cabo, en orden, las tres acciones de la definición 3.36:

i) Las componentes de z_i con $c \leq i \leq m$ sí existen, y son: $z_{1,}z_2,z_3,z_4$ y $z_{5,}$; por ello, sus índices se aumentan en 1.

$$i) \quad \mathbf{x} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} \text{ y } \mathbf{z} = \begin{pmatrix} z_1 = 0 \\ z_2 = 1 \\ z_3 = 0 \\ z_4 = 1 \\ z_5 = 0 \end{pmatrix} \longrightarrow \begin{pmatrix} \\ z_2 = 0 \\ z_3 = 1 \\ z_4 = 0 \\ z_5 = 1 \\ z_6 = 0 \end{pmatrix}$$

ii) Se inserta una componente en la posición $c = 1$ del vector $\mathbf{z}$, de modo que $z_c = 1$:

$$ii) \quad \begin{pmatrix} \\ z_2 = 0 \\ z_3 = 1 \\ z_4 = 0 \\ z_5 = 1 \\ z_6 = 0 \end{pmatrix} \longrightarrow \begin{pmatrix} z_1 = 1 \\ z_2 = 0 \\ z_3 = 1 \\ z_4 = 0 \\ z_5 = 1 \\ z_6 = 0 \end{pmatrix}$$

iii) Se ejecuta la siguiente asignación: $x_c = 0$.

$$iii) \quad \begin{pmatrix} x_1 = 0 \\ x_2 = 0 \\ x_3 = 0 \\ x_4 = 1 \\ x_5 = 0 \\ x_6 = 1 \\ x_7 = 0 \\ x_8 = 0 \end{pmatrix} = \mathcal{I}_1(\mathbf{x}) \text{ y } \begin{pmatrix} z_1 = 1 \\ z_2 = 0 \\ z_3 = 1 \\ z_4 = 0 \\ z_5 = 1 \\ z_6 = 0 \end{pmatrix} = \mathcal{I}_1(\mathbf{z})$$

Así, tenemos como resultado de la Inserción en $\mathbf{z}$ según $\mathbf{x}$: $\mathcal{I}_1(\mathbf{x}) = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$ y $\mathcal{I}_1(\mathbf{z}) = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \end{pmatrix}$.

Ejemplo 3.39 *Sean* $\mathbf{x} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \end{pmatrix}$ *y* $\mathbf{z} = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}$*; obtener la Inserción en* $\mathbf{z}$ *según* $\mathbf{x}$*.*

Como se puede ver, $\mathbf{x} \in A^7$, $n = 7$, $\sigma_n(\mathbf{x}) = 4$, $n - \sigma_n(\mathbf{x}) = 7 - 4 = 3$ y $\mathbf{z} \in A^3$, por lo que $m = n - \sigma_n(\mathbf{x})$, valores acordes a la definición 3.36; en este caso, el valor $c = 2$ cumple con la condición

$$c = \bigwedge_{j=1}^{n} (j \text{ tal que } x_j = 1)$$

Ahora, se llevan a cabo, en orden, las tres acciones de la definición 3.36:

Al realizar las tres acciones de la definición 3.36 se obtiene:

$$i) \quad \mathbf{x} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \end{pmatrix} \text{ y } \mathbf{z} = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \longrightarrow \begin{pmatrix} z_1 = 0 \\ \\ z_3 = 1 \\ z_4 = 0 \end{pmatrix}$$

$$ii) \quad \begin{pmatrix} z_1 = 1 \\ \\ z_3 = 1 \\ z_4 = 0 \end{pmatrix} \longrightarrow \begin{pmatrix} z_1 = 0 \\ z_2 = 1 \\ z_3 = 1 \\ z_4 = 0 \end{pmatrix}$$

$$iii) \quad \begin{pmatrix} x_1 = 0 \\ x_2 = 0 \\ x_3 = 0 \\ x_4 = 1 \\ x_5 = 1 \\ x_6 = 0 \\ x_7 = 1 \end{pmatrix} = \mathcal{I}_1(\mathbf{x}) \text{ y } \begin{pmatrix} z_1 = 0 \\ z_2 = 1 \\ z_3 = 1 \\ z_4 = 0 \end{pmatrix} = \mathcal{I}_1(\mathbf{z})$$

con lo que $\mathcal{I}_1(\mathbf{x}) = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \end{pmatrix}$ y $\mathcal{I}_1(\mathbf{z}) = \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$.

Ejemplo 3.40 *Sean* $\mathbf{x} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$ *y* $\mathbf{z} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}$*; obtener la Inserción en* $\mathbf{z}$ *según* $\mathbf{x}$*.*

En este caso $c = 4$; entonces, para obtener la Inserción en $\mathbf{z}$ según $\mathbf{x}$ se realiza lo siguiente:

$$i) \quad \mathbf{x} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix} \text{ y } \mathbf{z} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \longrightarrow \begin{pmatrix} z_1 = 1 \\ z_2 = 0 \\ z_3 = 0 \\ \\ z_5 = 1 \\ z_6 = 0 \end{pmatrix}$$

$$ii) \quad \begin{pmatrix} z_1 = 1 \\ z_2 = 0 \\ z_3 = 0 \\ \\ z_5 = 1 \\ z_6 = 0 \end{pmatrix} \longrightarrow \begin{pmatrix} z_1 = 1 \\ z_2 = 0 \\ z_3 = 0 \\ z_4 = 1 \\ z_5 = 1 \\ z_6 = 0 \end{pmatrix}$$

$$iii) \quad \begin{pmatrix} x_1 = 0 \\ x_2 = 0 \\ x_3 = 0 \\ x_4 = 0 \\ x_5 = 1 \\ x_6 = 0 \\ x_7 = 1 \\ x_8 = 1 \\ x_9 = 0 \end{pmatrix} = \mathcal{I}_1(\mathbf{x}) \text{ y } \begin{pmatrix} z_1 = 1 \\ z_2 = 0 \\ z_3 = 0 \\ z_4 = 1 \\ z_5 = 1 \\ z_6 = 0 \end{pmatrix} = \mathcal{I}_1(\mathbf{z})$$

con lo que $\mathcal{I}_1(\mathbf{x}) = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$ y $\mathcal{I}_1(\mathbf{z}) = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$.

Proposición 3.41 *Sea $A = \{0, 1\}$. Si $\mathbf{x}$ y $\mathbf{z}$ son dos vectores columna $\mathbf{x} \in A^n$ y $\mathbf{z} \in A^m$ con $\mathbf{x} \neq \mathbf{0}$, $\mathbf{x} \neq \mathbf{1}$, $\mathbf{z} \neq \mathbf{0}$, $\mathbf{z} \neq \mathbf{1}$, $n, m \in \mathbb{Z}^+$ y $m = n - \sigma_n(\mathbf{x})$, se cumple que al aplicar una vez la operación Inserción en $\mathbf{z}$ según $\mathbf{x}$, la dimensión del vector $\mathcal{I}_1(\mathbf{z})$ es $n - \sigma_n(\mathbf{x}) + 1$.*

Demostración.- Después de calcular el valor de c según la definición 3.36, los índices de las componentes z_i con $c \leq i \leq m$ se aumentan en 1; dado que el índice de la componente z_m se aumenta en 1, el índice de la última componente del vector

transformado $\mathcal{I}_1(\mathbf{z})$ es $m+1$, y por ello $\mathcal{I}_1(\mathbf{z}) \in A^{m+1}$. Entonces, la dimensión del vector $\mathcal{I}_1(\mathbf{z})$ es $m+1 = n - \sigma_n(\mathbf{x}) + 1$.■

Definición 3.42 *Sea* $A = \{0,1\}$*, y sean* $\mathbf{x}$ *y* $\mathbf{y}$ *dos vectores columna* $\mathbf{x} \in A^n$ *y* $\mathbf{y} \in A^m$ *con* $\mathbf{x} \neq \mathbf{0}$*,* $\mathbf{x} \neq \mathbf{1}$*,* $\mathbf{y} \neq \mathbf{0}$*,* $\mathbf{y} \neq \mathbf{1}$*,* $n, m \in \mathbb{Z}^+$ *y* $m = n - \sigma_n(\mathbf{x})$*. Se define la Expansión de* $\mathbf{y}$ *por* $\mathbf{x}$*, y se denota como* $\mathbf{y}|^{\mathbf{x}}$*, al vector que resulta, a partir del vector* $\mathbf{y}$*, de aplicar iteradamente la operación Inserción en* $\mathbf{y}$ *según* $\mathbf{x}$*, para todas y cada una de las componentes de* $\mathbf{x}$ *con valor* 1*.*

Ejemplo 3.43 *Sean* $\mathbf{x} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$ *y* $\mathbf{y} = \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$*; obtener* $\mathbf{y}|^{\mathbf{x}}$*, que es la Expansión de* $\mathbf{y}$ *por* $\mathbf{x}$*.*

De acuerdo con la definición 3.42, para obtener $\mathbf{y}|^{\mathbf{x}}$ es necesario llevar a cabo la Inserción en $\mathbf{y}$ según $\mathbf{x}$, para todas y cada una de las componentes de $\mathbf{x}$ con valor 1. Entonces, para $c = 1$:

$$\mathcal{I}_1(\mathbf{x}) = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix} \text{ y } \mathcal{I}_1(\mathbf{y}) = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$$

para $c = 4$:

$$\mathcal{I}_2(\mathbf{x}) = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix} \text{ y } \mathcal{I}_2(\mathbf{y}) = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ 0 \end{pmatrix}$$

para $c = 6$:

$$\mathcal{I}_3(\mathbf{x}) = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \text{ y } \mathcal{I}_3(\mathbf{y}) = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 0 \end{pmatrix}$$

y para $c = 7$:

$$\mathcal{I}_4(\mathbf{x}) = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \text{ y } \mathcal{I}_4(\mathbf{y}) = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 0 \end{pmatrix}$$

así que la Expansión de $\mathbf{y}$ por $\mathbf{x}$ es $\mathbf{y}|^{\mathbf{x}} = \mathcal{I}_4(\mathbf{y}) = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 0 \end{pmatrix}$.

Ejemplo 3.44 *Sean* $\mathbf{x} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}$ *y* $\mathbf{y} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$*; obtener* $\mathbf{y}|^{\mathbf{x}}$.

Para obtener $\mathbf{y}|^{\mathbf{x}}$ es necesario llevar a cabo la Inserción en $\mathbf{y}$ según $\mathbf{x}$ tres veces, puesto que $\mathbf{x}$ tiene 3 componentes con valor de 1; así, tenemos para $c = 1$, $c = 6$ y

$c = 9$, respectivamente:

$$\mathcal{I}_1(\mathbf{x}) = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} \text{ y } \mathcal{I}_1(\mathbf{y}) = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$$

$$\mathcal{I}_2(\mathbf{x}) = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} \text{ y } \mathcal{I}_2(\mathbf{y}) = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$$

$$\mathcal{I}_3(\mathbf{x}) = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \text{ y } \mathcal{I}_3(\mathbf{y}) = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$$

por lo que $\mathbf{y}|^{\mathbf{x}} = \mathcal{I}_3(\mathbf{y}) = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$.

Ejemplo 3.45 *Sean* $\mathbf{x} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{pmatrix}$ *y* $\mathbf{y} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}$*; obtener* $\mathbf{y}|^{\mathbf{x}}$.

En este caso se realizan 6 Inserciones en $\mathbf{y}$ según $\mathbf{x}$:

$$\mathcal{I}_1(\mathbf{x}) = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{pmatrix} \text{ y } \mathcal{I}_1(\mathbf{y}) = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \longrightarrow \mathcal{I}_2(\mathbf{x}) = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{pmatrix} \text{ y } \mathcal{I}_2(\mathbf{y}) = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

$$\longrightarrow \mathcal{I}_3(\mathbf{x}) = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{pmatrix} \text{ y } \mathcal{I}_3(\mathbf{y}) = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \longrightarrow \mathcal{I}_4(\mathbf{x}) = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{pmatrix} \text{ y } \mathcal{I}_4(\mathbf{y}) = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

$$\longrightarrow \mathcal{I}_5(\mathbf{x}) = \begin{pmatrix} 0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\1 \end{pmatrix} \text{ y } \mathcal{I}_5(\mathbf{y}) = \begin{pmatrix} 1\\0\\0\\1\\1\\1\\1\\0\\0\\1\\0 \end{pmatrix} \longrightarrow \mathcal{I}_6(\mathbf{x}) = \begin{pmatrix} 0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0 \end{pmatrix} \text{ y } \mathcal{I}_6(\mathbf{y}) = \begin{pmatrix} 1\\0\\0\\1\\1\\1\\1\\0\\0\\1\\0\\1 \end{pmatrix}$$

con lo que se obtiene $\mathbf{y}|^{\mathbf{x}} = \begin{pmatrix} 1\\0\\0\\1\\1\\1\\1\\0\\0\\1\\0\\1 \end{pmatrix}$.

Proposición 3.46 *Sea* $A = \{0,1\}$. *Si* $\mathbf{x}$ *y* $\mathbf{y}$ *son dos vectores columna* $\mathbf{x} \in A^n$ *y* $\mathbf{y} \in A^m$ *con* $\mathbf{x} \neq \mathbf{0}$, $\mathbf{x} \neq \mathbf{1}$, $\mathbf{y} \neq \mathbf{0}$, $\mathbf{y} \neq \mathbf{1}$, $n, m \in \mathbb{Z}^+$ *y* $m = n - \sigma_n(\mathbf{x})$, *se cumplen las siguientes dos afirmaciones:*

i) Para obtener $\mathbf{y}|^{\mathbf{x}}$, *se requiere aplicar* $\sigma_n(\mathbf{x})$ *veces la operación Inserción en* $\mathbf{y}$ *según* $\mathbf{x}$.

ii) La dimensión del vector $\mathbf{y}|^{\mathbf{x}}$ *es n; es decir,* $\mathbf{y}|^{\mathbf{x}} \in A^n$.

Demostración.-

i) De acuerdo con la Definición 3.36, cada vez que se aplica la operación Inserción en $\mathbf{y}$ según $\mathbf{x}$ se inserta en el vector $\mathbf{y}$ una componente con valor 1; lo cual está acorde con la proposición 3.41, porque la dimensión del vector $\mathcal{I}_1(\mathbf{y})$ es $m + 1$. Por otro lado, según la Definición 3.42, para obtener $\mathbf{y}|^{\mathbf{x}}$ se aplica iteradamente la operación Inserción en $\mathbf{y}$ según $\mathbf{x}$, para todas y cada una de las componentes de $\mathbf{x}$ con valor 1; pero según la proposición 3.16 el escalar $\sigma_n(\mathbf{x})$ representa precisamentre el número total de componentes con valor 1 que contiene el vector $\mathbf{x}$. Por lo tanto, se concluye que para obtener $\mathbf{y}|^{\mathbf{x}}$, se requiere aplicar $\sigma_n(\mathbf{x})$ veces la operación Inserción en $\mathbf{y}$ según $\mathbf{x}$.

ii) De acuerdo con la Definición 3.36, al aplicar una vez la operación Inserción en $\mathbf{y}$ según $\mathbf{x}$, se obtiene el vector transformado $\mathcal{I}_1(\mathbf{y})$ de dimensión $m+1 = (n-\sigma_n(\mathbf{x}))+1$. En la segunda iteración, el vector transformado $\mathcal{I}_2(\mathbf{y})$ será de dimensión $(m+1)+1 = m+2 = (n-\sigma_n(\mathbf{x}))+2$; así, al llegar a la última iteración, que es la iteración $\sigma_n(\mathbf{x})$ según

la parte i) de esta proposición, se tendrá que el vector transformado $\mathcal{I}_{\sigma_n(\mathbf{x})}(\mathbf{y})$, el cual es precisamente $\mathbf{y}|^{\mathbf{x}}$, tiene por dimensión el valor $m+\sigma_n(\mathbf{x}) = (n-\sigma_n(\mathbf{x}))+\sigma_n(\mathbf{x}) = n$; es decir, $\mathbf{y}|^{\mathbf{x}} \in A^n$.■

3.3. El vector de soporte para el nuevo modelo

Las definiciones y resultados preliminares de la sección previa, serán de gran utilidad en la presentación de algunos ejemplos que nos guíen hacia el nuevo concepto de vector de soporte, que es uno de los pilares conceptuales de las máquinas asociativas Alfa-Beta con soporte vectorial.

La secuencia de los ejemplos está diseñada de modo tal, que conlleva de manera natural hacia el concepto y definición del nuevo vector de soporte; para ello, se aprovecha el relevante hecho de que, bajo ciertas condiciones, la Expansión (la cual está basada en la operación de Inserción) es una especie de inversa de la Restricción (que a su vez se basa en la operación de Eliminación). Paso a paso, los ejemplos exhiben claramente las condiciones suficientes para que la Expansión sea, en efecto, inversa de la Restricción, y estas condiciones suficientes son plasmadas en el Teorema 3.56; a partir de ahí, surge claramente la necesidad de definir un vector de soporte que cumpla, para un conjunto fundamental dado, las condiciones suficientes de dicho Teorema.

La definición del nuevo vector de soporte, junto con dos transformadas originales introducidas en este trabajo de tesis, sirven como base fundamental del algoritmo principal con el que funcionan, en ambas fases (aprendizaje y recuperación de patrones), las máquinas asociativas Alfa-Beta con soporte vectorial.

3.3.1. Ejemplos ilustrativos

Ejemplo 3.47 *Sean* $\mathbf{x} \in A^n$ *y* $\mathbf{y^1} \in A^n$ *vectores columna de dimensión* $n = 8$*, donde* $\mathbf{x} = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \end{pmatrix}$ *y* $\mathbf{y^1} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{pmatrix}$*. Obtener* $\mathbf{y^1}|_{\mathbf{x}}$ *y* $\left(\mathbf{y^1}|_{\mathbf{x}}\right)|^{\mathbf{x}}$*.*

Al aplicar $\sigma_n(\mathbf{x}) = \mathbf{5}$ veces la operación Eliminación en $\mathbf{y^1}$ según $\mathbf{x}$, se obtiene $\mathbf{y^1}$ restringido por $\mathbf{x}$.

$$\mathcal{E}_1(\mathbf{x}) = \begin{pmatrix} 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \end{pmatrix} \text{ y } \mathcal{E}_1(\mathbf{y^1}) = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{pmatrix} \longrightarrow \mathcal{E}_2(\mathbf{x}) = \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \end{pmatrix} \text{ y } \mathcal{E}_2(\mathbf{y^1}) = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{pmatrix} \longrightarrow$$

$$\mathcal{E}_3(\mathbf{x}) = \begin{pmatrix}0\\1\\0\\1\\0\end{pmatrix} \text{ y } \mathcal{E}_3(\mathbf{y^1}) = \begin{pmatrix}1\\0\\1\\0\\1\end{pmatrix} \quad \longrightarrow \quad \mathcal{E}_4(\mathbf{x}) = \begin{pmatrix}0\\0\\1\\0\end{pmatrix} \text{ y } \mathcal{E}_4(\mathbf{y^1}) = \begin{pmatrix}1\\1\\0\\1\end{pmatrix} \quad \longrightarrow$$

$$\mathcal{E}_5(\mathbf{x}) = \begin{pmatrix}0\\0\\0\end{pmatrix} \text{ y } \mathcal{E}_5(\mathbf{y^1}) = \begin{pmatrix}1\\1\\1\end{pmatrix}$$

con lo que se obtiene la Restricción de $\mathbf{y^1}$ por $\mathbf{x}$, o sea, $\mathbf{y^1}\big|_{\mathbf{x}} = \begin{pmatrix}1\\1\\1\end{pmatrix}$.

De acuerdo con la definición 3.42, para obtener $\left(\mathbf{y^1}\big|_{\mathbf{x}}\right)\big|^{\mathbf{x}}$ es necesario llevar a cabo la Inserción en $\mathbf{y^1}\big|_{\mathbf{x}}$ según $\mathbf{x}$, para todas y cada una de las componentes de $\mathbf{x}$ con valor 1, es decir, la operación Inserción será efectuada $\sigma_n(\mathbf{x}) = \mathbf{5}$ veces.

$$\mathcal{I}_1(\mathbf{x}) = \begin{pmatrix}0\\0\\1\\1\\1\\0\\1\\0\end{pmatrix} \text{ y } \mathcal{I}_1(\mathbf{y^1}\big|_{\mathbf{x}}) = \begin{pmatrix}1\\1\\1\\1\end{pmatrix} \quad , \quad \mathcal{I}_2(\mathbf{x}) = \begin{pmatrix}0\\0\\0\\1\\1\\0\\1\\0\end{pmatrix} \text{ y } \mathcal{I}_2(\mathbf{y^1}\big|_{\mathbf{x}}) = \begin{pmatrix}1\\1\\1\\1\\1\end{pmatrix}$$

$$\mathcal{I}_3(\mathbf{x}) = \begin{pmatrix}0\\0\\0\\0\\1\\0\\1\\0\end{pmatrix} \text{ y } \mathcal{I}_3(\mathbf{y^1}\big|_{\mathbf{x}}) = \begin{pmatrix}1\\1\\1\\1\\1\\1\end{pmatrix} \quad , \quad \mathcal{I}_4(\mathbf{x}) = \begin{pmatrix}0\\0\\0\\0\\0\\0\\1\\0\end{pmatrix} \text{ y } \mathcal{I}_4(\mathbf{y^1}\big|_{\mathbf{x}}) = \begin{pmatrix}1\\1\\1\\1\\1\\1\\1\\1\end{pmatrix}$$

$$\mathcal{I}_5(\mathbf{x}) = \begin{pmatrix}0\\0\\0\\0\\0\\0\\0\\0\end{pmatrix} \text{ y } \mathcal{I}_5(\mathbf{y^1}\big|_{\mathbf{x}}) = \begin{pmatrix}1\\1\\1\\1\\1\\1\\1\\1\end{pmatrix}$$

con lo que se obtiene la Expansión de $\mathbf{y^1}\big|_{\mathbf{x}}$ por $\mathbf{x}$: $\left(\mathbf{y^1}\big|_{\mathbf{x}}\right)\big|^{\mathbf{x}} = \begin{pmatrix}1\\1\\1\\1\\1\\1\\1\\1\end{pmatrix}$.

Al comparar el vector obtenido mediante la Expansión de $\mathbf{y^1}\big|_{\mathbf{x}}$ por $\mathbf{x}$ contra el vector original $\mathbf{y^1}$, se puede observar que existen 5 posiciones incorrectamente recuperadas, las cuales han sido encerradas en rectángulos en la siguiente expresión:

$$\mathbf{y^1} = \begin{pmatrix}0\\1\\0\\0\\0\\1\\0\\1\end{pmatrix} \qquad \left(\mathbf{y^1}\big|_{\mathbf{x}}\right)\big|^{\mathbf{x}} = \begin{pmatrix}\boxed{1}\\1\\\boxed{1}\\\boxed{1}\\\boxed{1}\\1\\\boxed{1}\\1\end{pmatrix}$$

Ejemplo 3.48 *Sean* $\mathbf{x} \in A^n$ *y* $\mathbf{y^2} \in A^n$ *vectores columna de dimensión* $n = 8$*, donde* $\mathbf{x} = \begin{pmatrix}1\\0\\1\\1\\1\\0\\1\\0\end{pmatrix}$ *y* $\mathbf{y^2} = \begin{pmatrix}0\\1\\0\\0\\1\\0\\0\\1\end{pmatrix}$. *Obtener* $\mathbf{y^2}\big|_{\mathbf{x}}$ *y* $\left(\mathbf{y^2}\big|_{\mathbf{x}}\right)\big|^{\mathbf{x}}$.

Al aplicar $\sigma_n(\mathbf{x}) = \mathbf{5}$ veces la operación Eliminación en $\mathbf{y^2}$ según $\mathbf{x}$, se obtiene la Restricción de $\mathbf{y^2}$ por $\mathbf{x}$, o sea, $\mathbf{y^2}\big|_{\mathbf{x}} = \begin{pmatrix}1\\0\\1\end{pmatrix}$.

De acuerdo con la definición 3.42, para obtener $\left(\mathbf{y^2}\big|_{\mathbf{x}}\right)\big|^{\mathbf{x}}$ es necesario llevar a cabo la Inserción en $\mathbf{y^2}\big|_{\mathbf{x}}$ según $\mathbf{x}$, para todas y cada una de las componentes de $\mathbf{x}$ con valor 1, es decir, la operación Inserción será efectuada $\sigma_n(\mathbf{x}) = \mathbf{5}$ veces, con lo que se obtiene la Expansión de $\mathbf{y^2}\big|_{\mathbf{x}}$ por $\mathbf{x}$: $\left(\mathbf{y^2}\big|_{\mathbf{x}}\right)\big|^{\mathbf{x}} = \begin{pmatrix}1\\1\\1\\1\\1\\0\\1\\1\end{pmatrix}$.

Al comparar el vector obtenido mediante la Expansión de $\mathbf{y^2}\big|_{\mathbf{x}}$ por $\mathbf{x}$ contra el vector original $\mathbf{y^2}$, se puede observar que existen 4 posiciones incorrectamente recuperadas, las cuales han sido encerradas en rectángulos en la siguiente expresión:

$$\mathbf{y^2} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} \qquad \left(\mathbf{y^2}\big|_{\mathbf{x}}\right)\Big|^{\mathbf{x}} = \begin{pmatrix} \boxed{1} \\ 1 \\ \boxed{1} \\ \boxed{1} \\ 1 \\ 0 \\ \boxed{1} \\ 1 \end{pmatrix}$$

Ejemplo 3.49 *Sean* $\mathbf{x} \in A^n$ *y* $\mathbf{y^3} \in A^n$ *vectores columna de dimensión* $n = 8$*, donde* $\mathbf{x} = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \end{pmatrix}$ *y* $\mathbf{y^3} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \end{pmatrix}$*. Obtener* $\mathbf{y^3}\big|_{\mathbf{x}}$ *y* $\left(\mathbf{y^3}\big|_{\mathbf{x}}\right)\Big|^{\mathbf{x}}$.

Al aplicar $\sigma_n(\mathbf{x}) = \mathbf{5}$ veces la operación Eliminación en $\mathbf{y^3}$ según $\mathbf{x}$, se obtiene la Restricción de $\mathbf{y^3}$ por $\mathbf{x}$, o sea, $\mathbf{y^3}\big|_{\mathbf{x}} = \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix}$.

De acuerdo con la definición 3.42, para obtener $\left(\mathbf{y^3}\big|_{\mathbf{x}}\right)\Big|^{\mathbf{x}}$ es necesario llevar a cabo la Inserción en $\mathbf{y^3}\big|_{\mathbf{x}}$ según $\mathbf{x}$, para todas y cada una de las componentes de $\mathbf{x}$ con valor 1, es decir, la operación Inserción será efectuada $\sigma_n(\mathbf{x}) = \mathbf{5}$ veces, con lo que se obtiene la Expansión de $\mathbf{y^3}\big|_{\mathbf{x}}$ por $\mathbf{x}$: $\left(\mathbf{y^3}\big|_{\mathbf{x}}\right)\Big|^{\mathbf{x}} = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$.

Al comparar el vector obtenido mediante la Expansión de $\mathbf{y^3}\big|_{\mathbf{x}}$ por $\mathbf{x}$ contra el vector original $\mathbf{y^3}$, se puede observar que existen 3 posiciones incorrectamente recuperadas, las cuales han sido encerradas en rectángulos en la siguiente expresión:

$$\mathbf{y^3} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \end{pmatrix} \qquad \left(\mathbf{y^3}\big|_{\mathbf{x}}\right)\big|^{\mathbf{x}} = \begin{pmatrix} 1 \\ 0 \\ \boxed{1} \\ \boxed{1} \\ \boxed{1} \\ 1 \\ 1 \\ 1 \end{pmatrix}$$

Ejemplo 3.50 *Sean* $\mathbf{x} \in A^n$ *y* $\mathbf{y^4} \in A^n$ *vectores columna de dimensión* $n = 8$*, donde* $\mathbf{x} = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \end{pmatrix}$ *y* $\mathbf{y^4} = \begin{pmatrix} 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$*. Obtener* $\mathbf{y^4}\big|_{\mathbf{x}}$ *y* $\left(\mathbf{y^4}\big|_{\mathbf{x}}\right)\big|^{\mathbf{x}}$*.*

Al aplicar $\sigma_n(\mathbf{x}) = \mathbf{5}$ veces la operación Eliminación en $\mathbf{y^4}$ según $\mathbf{x}$, se obtiene la Restricción de $\mathbf{y^4}$ por $\mathbf{x}$, o sea, $\mathbf{y^4}\big|_{\mathbf{x}} = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}$.

De acuerdo con la definición 3.42, para obtener $\left(\mathbf{y^4}\big|_{\mathbf{x}}\right)\big|^{\mathbf{x}}$ es necesario llevar a cabo la Inserción en $\mathbf{y^4}\big|_{\mathbf{x}}$ según $\mathbf{x}$, para todas y cada una de las componentes de $\mathbf{x}$ con valor 1, es decir, la operación Inserción será efectuada $\sigma_n(\mathbf{x}) = \mathbf{5}$ veces.con lo que se obtiene la Expansión de $\mathbf{y^4}\big|_{\mathbf{x}}$ por $\mathbf{x}$: $\left(\mathbf{y^4}\big|_{\mathbf{x}}\right)\big|^{\mathbf{x}} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 0 \end{pmatrix}$.

Al comparar el vector obtenido mediante la Expansión de $\mathbf{y^4}\big|_{\mathbf{x}}$ por $\mathbf{x}$ contra el vector original $\mathbf{y^4}$, se puede observar que existen 2 posiciones incorrectamente recuperadas, las cuales han sido encerradas en rectángulos en la siguiente expresión:

$$\mathbf{y^4} = \begin{pmatrix} 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix} \qquad \left(\mathbf{y^4}\big|_{\mathbf{x}}\right)\big|^{\mathbf{x}} = \begin{pmatrix} \boxed{1} \\ 1 \\ 1 \\ 1 \\ \boxed{1} \\ 1 \\ 1 \\ 0 \end{pmatrix}$$

Ejemplo 3.51 *Sean* $\mathbf{x} \in A^n$ *y* $\mathbf{y^5} \in A^n$ *vectores columna de dimensión* $n = 8$*, donde* $\mathbf{x} = \begin{pmatrix}1\\0\\1\\1\\1\\0\\1\\0\end{pmatrix}$ *y* $\mathbf{y^5} = \begin{pmatrix}1\\1\\1\\0\\1\\0\\1\\1\end{pmatrix}$*. Obtener* $\mathbf{y^5}\big|_{\mathbf{x}}$ *y* $\left(\mathbf{y^5}\big|_{\mathbf{x}}\right)\big|^{\mathbf{x}}$*.*

Al aplicar $\sigma_n(\mathbf{x}) = \mathbf{5}$ veces la operación Eliminación en $\mathbf{y^5}$ según $\mathbf{x}$, se obtiene la Restricción de $\mathbf{y^5}$ por $\mathbf{x}$, o sea, $\mathbf{y^5}\big|_{\mathbf{x}} = \begin{pmatrix}1\\0\\1\end{pmatrix}$.

De acuerdo con la definición 3.42, para obtener $\left(\mathbf{y^5}\big|_{\mathbf{x}}\right)\big|^{\mathbf{x}}$ es necesario llevar a cabo la Inserción en $\mathbf{y^5}\big|_{\mathbf{x}}$ según $\mathbf{x}$, para todas y cada una de las componentes de $\mathbf{x}$ con valor 1, es decir, la operación Inserción será efectuada $\sigma_n(\mathbf{x}) = \mathbf{5}$ veces, con lo que se obtiene la Expansión de $\mathbf{y^5}\big|_{\mathbf{x}}$ por $\mathbf{x}$: $\left(\mathbf{y^5}\big|_{\mathbf{x}}\right)\big|^{\mathbf{x}} = \begin{pmatrix}1\\1\\1\\1\\1\\0\\1\\1\end{pmatrix}$.

Al comparar el vector obtenido mediante la Expansión de $\mathbf{y^5}\big|_{\mathbf{x}}$ por $\mathbf{x}$ contra el vector original $\mathbf{y^5}$, se puede observar que existe 1 posición erróneamente recuperada, la cual ha sido encerrada en un rectángulo en la siguiente expresión:

$$\mathbf{y^5} = \begin{pmatrix}1\\1\\1\\0\\1\\0\\1\\1\end{pmatrix} \qquad \left(\mathbf{y^5}\big|_{\mathbf{x}}\right)\big|^{\mathbf{x}} = \begin{pmatrix}1\\1\\1\\\boxed{1}\\1\\0\\1\\1\end{pmatrix}$$

Ejemplo 3.52 *Sean* $\mathbf{x} \in A^n$ *y* $\mathbf{y^6} \in A^n$ *vectores columna de dimensión* $n = 8$*, donde* $\mathbf{x} = \begin{pmatrix}1\\0\\1\\1\\1\\0\\1\\0\end{pmatrix}$ *y* $\mathbf{y^6} = \begin{pmatrix}1\\1\\1\\1\\1\\0\\1\\0\end{pmatrix}$*. Obtener* $\mathbf{y^6}\big|_{\mathbf{x}}$ *y* $\left(\mathbf{y^6}\big|_{\mathbf{x}}\right)\big|^{\mathbf{x}}$*.*

Al aplicar $\sigma_n(\mathbf{x}) = \mathbf{5}$ veces la operación Eliminación en $\mathbf{y^6}$ según $\mathbf{x}$, se obtiene la Restricción de $\mathbf{y^6}$ por $\mathbf{x}$, o sea, $\mathbf{y^6}\big|_{\mathbf{x}} = \begin{pmatrix}1\\0\\0\end{pmatrix}$.

De acuerdo con la definición 3.42, para obtener $\left(\mathbf{y^6}\big|_{\mathbf{x}}\right)\big|^{\mathbf{x}}$ es necesario llevar a cabo la Inserción en $\mathbf{y^6}\big|_{\mathbf{x}}$ según $\mathbf{x}$, para todas y cada una de las componentes de $\mathbf{x}$ con valor 1, es decir, la operación Inserción será efectuada $\sigma_n(\mathbf{x}) = \mathbf{5}$ veces, con lo que se obtiene la Expansión de $\mathbf{y^6}\big|_{\mathbf{x}}$ por $\mathbf{x}$: $\left(\mathbf{y^6}\big|_{\mathbf{x}}\right)\big|^{\mathbf{x}} = \begin{pmatrix}1\\1\\1\\1\\1\\0\\1\\0\end{pmatrix}$.

Al comparar el vector obtenido mediante la Expansión de $\mathbf{y^6}\big|_{\mathbf{x}}$ por $\mathbf{x}$ contra el vector original $\mathbf{y^6}$, se puede observar que todas las posiciones son correctamente recuperadas:

$$\mathbf{y^6} = \begin{pmatrix}1\\1\\1\\1\\1\\0\\1\\0\end{pmatrix} \qquad \left(\mathbf{y^6}\big|_{\mathbf{x}}\right)\big|^{\mathbf{x}} = \begin{pmatrix}1\\1\\1\\1\\1\\0\\1\\0\end{pmatrix}$$

3.3.2. Análisis de los ejemplos ilustrativos

Si consideramos que la aplicación de la Expansión de un vector a la Restricción del mismo según otro vector es una *recuperación*, el análisis de lo que sucedió en el ejemplo 3.47 nos arroja que 5 de las componentes, encerradas en rectángulos en la figura siguiente, fueron recuperadas incorrectamente.

$$\mathbf{x} = \begin{pmatrix}1\\0\\1\\1\\1\\0\\1\\0\end{pmatrix} \quad \mathbf{y^1} = \begin{pmatrix}0\\1\\0\\0\\0\\1\\0\\1\end{pmatrix} \quad \mathbf{y^1}\big|_{\mathbf{x}} = \begin{pmatrix}1\\1\\1\end{pmatrix} \quad \left(\mathbf{y^1}\big|_{\mathbf{x}}\right)\big|^{\mathbf{x}} = \begin{pmatrix}\boxed{1}\\1\\\boxed{1}\\\boxed{1}\\\boxed{1}\\1\\\boxed{1}\\1\end{pmatrix}$$

Veamos qué se observa en las posiciones de esas componentes:

- Para la posición 1, $x_1 = 1$ y $y^1_1 = 0$.

- Para la posición 3, $x_3 = 1$ y $y_3^1 = 0$.
- Para la posición 4, $x_4 = 1$ y $y_4^1 = 0$.
- Para la posición 5, $x_5 = 1$ y $y_5^1 = 0$.
- Para la posición 7, $x_7 = 1$ y $y_7^1 = 0$.

Por otro lado, en las posiciones donde las componentes del vector $\mathbf{x}$ tienen valor 0, la recuperación fue correcta. Se observa también que estas posiciones coinciden con las posiciones del vector $\mathbf{y}^1$ de donde se formó el vector $\mathbf{y^1}|_{\mathbf{x}}$.

Los resultados del análisis previo inspiran el primer lema de esta sección.

Lema 3.53 *Sea $A = \{0, 1\}$, y sean $\mathbf{x}$ y $\mathbf{y}$ dos vectores columna $\mathbf{x} \in A^n$ y $\mathbf{y} \in A^n$ con $\mathbf{x} \neq \mathbf{0}$, $\mathbf{x} \neq \mathbf{1}$, $\mathbf{y} \neq \mathbf{0}$, $\mathbf{y} \neq \mathbf{1}$ y $n \in \mathbb{Z}^+$. Si $x_i = 1$ y $y_i = 0$ con $i \in \mathbb{Z}^+$ tal que $1 \leq i \leq n$, entonces $[(\mathbf{y}|_{\mathbf{x}})|^{\mathbf{x}}]_i \neq y_i$; es decir, y_i no se recupera de manera correcta.*

Demostración.- De acuerdo con la Definición 3.21, en el proceso de calcular $\mathbf{y}|_{\mathbf{x}}$, la Restricción de $\mathbf{y}$ según $\mathbf{x}$, al ser $x_i = 1$ se eliminan ambas componentes, x_i y y_i. Esto significa que en el vector resultante, $\mathbf{y}|_{\mathbf{x}}$, ya no aparece el valor 0 que correspondía a y_i, de modo que al realizar la operación de Inserción para el índice i en el proceso de calcular $(\mathbf{y}|_{\mathbf{x}})|^{\mathbf{x}}$, la Expansión de $\mathbf{y}|_{\mathbf{x}}$ según $\mathbf{x}$, donde había un valor 0 se coloca un valor 1, de acuerdo con la Definición 3.36. Por ello, al ser $y_i = 0$ y $[(\mathbf{y}|_{\mathbf{x}})|^{\mathbf{x}}]_i = 1$, se tiene que $[(\mathbf{y}|_{\mathbf{x}})|^{\mathbf{x}}]_i \neq y_i$; es decir, y_i no se recupera de manera correcta.■

A continuación se pesentan los vectores correspondientes al ejemplo 3.48 donde, de nuevo como en todos los ejemplos, las componentes recuperadas incorrectamente se encierran en rectángulos.

$$\mathbf{x} = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \end{pmatrix} \quad \mathbf{y^2} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} \quad \mathbf{y^2}|_{\mathbf{x}} = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} \quad (\mathbf{y^2}|_{\mathbf{x}})|^{\mathbf{x}} = \begin{pmatrix} \boxed{1} \\ 1 \\ \boxed{1} \\ \boxed{1} \\ 1 \\ 0 \\ \boxed{1} \\ 1 \end{pmatrix}$$

Al igual que en el ejemplo 3.47 , en las posiciones $i = 1, 3, 4, 7$ de este ejemplo 3.48 donde la recuperación es incorrecta, se cumple que $x_i = 1$ y $y_i^2 = 0$, acorde con el Lema 3.53.

Sin embargo, en las posiciones $i = 2, 5, 6, 8$ donde la recuperación es correcta, se observa algo novedoso; mientras que en las posiciones $i = 2, 6, 8$ se cumple lo mismo que en el ejemplo 3.47; es decir, que $x_2 = x_6 = x_8 = 0$, la posición $i = 5$ es especial, porque $x_5 \neq 0$, pero se cumple que $x_5 = 1$ y también $y_5^2 = 1$.

Veamos qué sucede con los resultados del ejemplo 3.49:

$$\mathbf{x} = \begin{pmatrix}1\\0\\1\\1\\1\\0\\1\\0\end{pmatrix} \quad \mathbf{y^3} = \begin{pmatrix}1\\0\\0\\0\\0\\1\\1\\1\end{pmatrix} \quad \mathbf{y^3}\big|_{\mathbf{x}} = \begin{pmatrix}0\\1\\1\end{pmatrix} \quad (\mathbf{y^3}\big|_{\mathbf{x}})|^{\mathbf{x}} = \begin{pmatrix}1\\0\\\boxed{1}\\\boxed{1}\\\boxed{1}\\1\\1\\1\end{pmatrix}$$

Aquí sólo hay tres componentes recuperadas de manera incorrecta, las cuales cumplen la misma condición de los ejemplos 3.47 y 3.48 (Lema 3.53):

- Para la posición 3, $x_3 = 1$ y $y_3^3 = 0$.
- Para la posición 4, $x_4 = 1$ y $y_4^3 = 0$.
- Para la posición 5, $x_5 = 1$ y $y_5^3 = 0$.

De las posiciones donde la recuperación fue correcta, podemos distinguir dos casos; uno, donde la posición coincide con alguna de las posiciones del vector $\mathbf{y^3}$ a partir de las cuales se formó el vector $\mathbf{y^3}\big|_{\mathbf{x}}$, y que son:

- Para la posición 2, $x_2 = 0$ y $y_2^3 = 0$.
- Para la posición 6, $x_6 = 0$ y $y_6^3 = 1$.
- Para la posición 8, $x_8 = 0$ y $y_8^3 = 1$.

Obsérvese que para este grupo de posiciones, no importa el valor de la componente en el vector $\mathbf{y^3}$, sino el hecho de que la componente en el vector $\mathbf{x}$ sea 0. Pero además, estas posiciones coinciden con aquellas posiciones, tanto del vector $\mathbf{x}$ como del $\mathbf{y^3}$, a partir de las cuales se formó el vector $\mathbf{y^3}\big|_{\mathbf{x}}$.

El segundo grupo de posiciones correspode a las que no contribuyen en la formación del vector $\mathbf{y^3}\big|_{\mathbf{x}}$, las cuales son:

- Para la posición 1, $x_1 = 1$ y $y_1^3 = 1$.
- Para la posición 7, $x_7 = 1$ y $y_7^3 = 1$.

Obsérvese que en todos los casos de este segundo grupo, el valor de las componentes es 1 en ambos vectores, $\mathbf{x}$ y $\mathbf{y^3}$.

Con el material proporcionado por el análisis de los ejemplos anteriores, ya se van perfilando ciertas conclusiones, que se confirmarán con el análisis del ejemplo 3.50 a continuación.

$$\mathbf{x} = \begin{pmatrix}1\\0\\1\\1\\1\\0\\1\\0\end{pmatrix} \quad \mathbf{y^4} = \begin{pmatrix}0\\1\\1\\1\\0\\1\\1\\0\end{pmatrix} \quad \mathbf{y^4}|_{\mathbf{x}} = \begin{pmatrix}1\\1\\0\end{pmatrix} \quad (\mathbf{y^4}|_{\mathbf{x}})|^{\mathbf{x}} = \begin{pmatrix}\boxed{1}\\1\\1\\1\\\boxed{1}\\1\\1\\0\end{pmatrix}$$

Se confirma que las posiciones $i = 2, 6, 8$ donde $x_i = 0$ son condiciones de no importa, dado que se recuperan de manera correcta, sin importar el valor de y_i^4.

Al parecer, las posiciones relevantes son aquellas donde $x_i = 1$, y se enlistan a continuación, en dos grupos claramente distinguibles.

Primer grupo, donde las recuperaciones son incorrectas (Lema 3.53):

- Para la posición 1, $x_1 = 1$ y $y_1^4 = 0$.
- Para la posición 5, $x_5 = 1$ y $y_5^4 = 0$.

Segundo grupo, donde las recuperaciones son correctas:

- Para la posición 3, $x_3 = 1$ y $y_3^4 = 1$.
- Para la posición 4, $x_4 = 1$ y $y_4^4 = 1$.
- Para la posición 7, $x_7 = 1$ y $y_7^4 = 1$.

Estas consideraciones dan pie al enunciado y demostración de dos lemas adicionales.

Lema 3.54 *Sea $A = \{0, 1\}$, y sean $\mathbf{x}$ y $\mathbf{y}$ dos vectores columna $\mathbf{x} \in A^n$ y $\mathbf{y} \in A^n$ con $\mathbf{x} \neq \mathbf{0}$, $\mathbf{x} \neq \mathbf{1}$, $\mathbf{y} \neq \mathbf{0}$, $\mathbf{y} \neq \mathbf{1}$ y $n \in \mathbb{Z}^+$. Si $x_i = 0$ con $i \in \mathbb{Z}^+$ tal que $1 \leq i \leq n$, entonces $[(\mathbf{y}|_{\mathbf{x}})|^{\mathbf{x}}]_i = y_i$; es decir, y_i se recupera de manera correcta.*

Demostración.- De acuerdo con la Definición 3.21, en el proceso de calcular $\mathbf{y}|_{\mathbf{x}}$, la Restricción de $\mathbf{y}$ según $\mathbf{x}$, al ser $x_i = 0$, el valor de y_i permanece en el vector $\mathbf{y}|_{\mathbf{x}}$, sin importar que $y_i = 0$ o que $y_i = 1$, aunque el índice de compnente cambia según la definición del proceso de obtener $\mathbf{y}|_{\mathbf{x}}$, la Restricción de $\mathbf{y}$ según $\mathbf{x}$. Sin embargo, al realizar la operación de Inserción para el índice i en el proceso de calcular $(\mathbf{y}|_{\mathbf{x}})|^{\mathbf{x}}$, la componente vuelve a tener el índice original i. Esto significa que al realizar la Expansión de $\mathbf{y}|_{\mathbf{x}}$ según $\mathbf{x}$, donde había un valor 0 se coloca un valor 0, y donde había un valor 1 se coloca un valor 1, de acuerdo con la Definición 3.36. Por ello se tiene que $[(\mathbf{y}|_{\mathbf{x}})|^{\mathbf{x}}]_i = y_i$; es decir, y_i se recupera de manera correcta.■

Lema 3.55 *Sea $A = \{0, 1\}$, y sean $\mathbf{x}$ y $\mathbf{y}$ dos vectores columna $\mathbf{x} \in A^n$ y $\mathbf{y} \in A^n$ con $\mathbf{x} \neq \mathbf{0}$, $\mathbf{x} \neq \mathbf{1}$, $\mathbf{y} \neq \mathbf{0}$, $\mathbf{y} \neq \mathbf{1}$ y $n \in \mathbb{Z}^+$. Si $x_i = 1$ y $y_i = 1$ con $i \in \mathbb{Z}^+$ tal que $1 \leq i \leq n$, entonces $[(\mathbf{y}|_{\mathbf{x}})|^{\mathbf{x}}]_i = y_i = 1$; es decir, y_i se recupera de manera correcta.*

Demostración.- De acuerdo con la Definición 3.21, en el proceso de calcular $\mathbf{y}|_{\mathbf{x}}$, la Restricción de $\mathbf{y}$ según $\mathbf{x}$, al ser $x_i = 1$ se eliminan ambas componentes, x_i y y_i. Esto significa que en el vector resultante, $\mathbf{y}|_{\mathbf{x}}$, no obstante que ya no aparece el valor 1 que correspondía a y_i, al realizar la operación de Inserción para el índice i en el proceso de calcular $(\mathbf{y}|_{\mathbf{x}})|^{\mathbf{x}}$, la Expansión de $\mathbf{y}|_{\mathbf{x}}$ según $\mathbf{x}$, de cualquier modo se coloca un valor 1, de acuerdo con la Definición 3.36. Por ello, al ser $y_i = 1$ y $[(\mathbf{y}|_{\mathbf{x}})|^{\mathbf{x}}]_i = 1$, se tiene que $[(\mathbf{y}|_{\mathbf{x}})|^{\mathbf{x}}]_i = y_i$; es decir, y_i se recupera de manera correcta.■

Los resultados del ejemplo 3.51 confirman lo anterior:

$$\mathbf{x} = \begin{pmatrix}1\\0\\1\\1\\1\\0\\1\\0\end{pmatrix} \quad \mathbf{y^5} = \begin{pmatrix}1\\1\\1\\0\\1\\0\\1\\1\end{pmatrix} \quad \mathbf{y^5}|_{\mathbf{x}} = \begin{pmatrix}1\\0\\1\end{pmatrix} \quad (\mathbf{y^5}|_{\mathbf{x}})|^{\mathbf{x}} = \begin{pmatrix}1\\1\\1\\\boxed{1}\\1\\0\\1\\1\end{pmatrix}$$

En efecto, la única componente donde la recuperación fue incorrecta es la que tiene índice 4, donde $x_4 = 1$ y $y_4^5 = 0$ (Lema 3.53). En congruencia con lo analizado anteriormente, las recuperaciones correctas, además de ocurrir en las posiciones donde las componentes del vector $\mathbf{x}$ son cero (Lema 3.54), se dan en las siguientes posiciones (Lema 3.55):

- Para la posición 1, $x_1 = 1$ y $y_1^5 = 1$.
- Para la posición 3, $x_3 = 1$ y $y_3^5 = 1$.
- Para la posición 5, $x_5 = 1$ y $y_5^5 = 1$.
- Para la posición 7, $x_7 = 1$ y $y_7^5 = 1$.

Los resultados del ejemplo 3.52, donde todas las componentes son recuperadas de manera correcta permiten, como consecuencia de los análisis previos, inducir condiciones suficientes para la recuperación correcta.

$$\mathbf{x} = \begin{pmatrix}1\\0\\1\\1\\1\\0\\1\\0\end{pmatrix} \quad \mathbf{y^6} = \begin{pmatrix}1\\1\\1\\1\\1\\0\\1\\0\end{pmatrix} \quad \mathbf{y^6}|_{\mathbf{x}} = \begin{pmatrix}1\\0\\0\end{pmatrix} \quad (\mathbf{y^6}|_{\mathbf{x}})|^{\mathbf{x}} = \begin{pmatrix}1\\1\\1\\1\\1\\0\\1\\0\end{pmatrix}$$

En efecto, se observa que en las posiciones $i = 2, 6, 8$ donde $x_i = 0$ la recuperación es correcta, sin condiciones adicionales; por otro lado, si $x_i = 1$, la condición que se

induce para la recuperación correcta es que $y_i^6 = 1$, como se observa en las posiciones $i = 1, 3, 4, 5, 7$.

Los análisis anteriores se resumen en el siguiente Teorema:

Teorema 3.56 *Sea $A = \{0, 1\}$, y sean $\mathbf{x}$ y $\mathbf{y}$ dos vectores columna $\mathbf{x} \in A^n$ y $\mathbf{y} \in A^n$ con $\mathbf{x} \neq \mathbf{0}$, $\mathbf{x} \neq \mathbf{1}$, $\mathbf{y} \neq \mathbf{0}$, $\mathbf{y} \neq \mathbf{1}$ y $n \in \mathbb{Z}^+$. Si $i \in \mathbb{Z}^+$ tal que $1 \leq i \leq n$, entonces el único caso para el que y_i no se recupera de manera correcta, y por ello $[(\mathbf{y}|_{\mathbf{x}})|^{\mathbf{x}}]_i \neq y_i$, es cuando $x_i = 1$ y $y_i = 0$.*

Demostración.- Hay cuatro casos mutuamente excluyentes:

CASO 1 $x_i = 0$ y $y_i = 0$: Según el Lema 3.54, $[(\mathbf{y}|_{\mathbf{x}})|^{\mathbf{x}}]_i = y_i$; es decir, y_i se recupera de manera correcta.

CASO 2 $x_i = 0$ y $y_i = 1$: Según el Lema 3.54, $[(\mathbf{y}|_{\mathbf{x}})|^{\mathbf{x}}]_i = y_i$; es decir, y_i se recupera de manera correcta.

CASO 3 $x_i = 1$ y $y_i = 1$: Según el Lema 3.55, $[(\mathbf{y}|_{\mathbf{x}})|^{\mathbf{x}}]_i = y_i$; es decir, y_i se recupera de manera correcta.

CASO 4 $x_i = 1$ y $y_i = 0$: Según el Lema 3.53, $[(\mathbf{y}|_{\mathbf{x}})|^{\mathbf{x}}]_i \neq y_i$; es decir, y_i no se recupera de manera correcta.

Por lo tanto, el único caso para el que y_i no se recupera de manera correcta, y por ello $[(\mathbf{y}|_{\mathbf{x}})|^{\mathbf{x}}]_i \neq y_i$, es cuando $x_i = 1$ y $y_i = 0$.■

3.3.3. Definición del nuevo vector de soporte, ejemplos y resultados

El análisis de los ejemplos ilustrativos y los resultados expresados en el Teorema 3.56, proporcionan los cimientos conceptuales sobre los que descansa el concepto del nuevo vector de soporte, que a su vez sirve de base teórica fundamental del nuevo modelo de máquinas asociativas Alfa-Beta con soporte vectorial.

La idea central que permite generar una definición de vector de soporte útil, está dada por el Teorema 3.56. El vector $\mathbf{x}$, según el cual se realiza tanto la Restricción como la Expansión del vector $\mathbf{y}$, se perfila como el vector de soporte; y el vector $\mathbf{y}$, sobre el que recaen los efectos tanto de la Restricción como de la Expansión, toma el papel de uno de los vectores del conjunto fundamental para las máquinas asociativas Alfa-Beta con soporte vectorial.

Si en un problema dado el conjunto fundamental es $\{(\mathbf{x}^\mu, \mathbf{x}^\mu) \mid \mu = 1, 2, ..., p\}$ con $\mathbf{x}^\mu \in A^n \ \forall \mu \in \{1, 2, ..., p\}$, $A = \{0, 1\}$, $n \in \mathbb{Z}^+$, $p \in \mathbb{Z}^+$, y $1 < p \leq 2^n$, lo primero que debe exigirse a un sistema de reconocimiento de patrones es que el conjunto fundamental completo sea recuperado de manera correcta. A fin de lograr que el nuevo modelo de máquinas asciativas Alfa-Beta con soporte vectorial recupere correctamente todos y cada uno de los patrones del conjunto fundamental, basta con asegurar que nunca ocurra el CASO 4 del Teorema 3.56; para ello, el vector de soporte $\mathbf{S}$ debe definirse de modo tal, que para ningún $\mu \in \{1, 2, ..., p\}$ y para ninguna $i \in \{1, 2, ..., n\}$ ocurra el caso en el que $S_i = 1$ y $x_i^\mu = 0$.

Lo anterior se puede lograr aplicando una regla simple: el vector de soporte $\mathbf{S}$ no podrá tener un 1 en la componente de posición i, si en esa posición al menos uno de los patrones del conjunto fundamental tiene un 0; en otras palabras, S_i tendrá valor 1 si y sólo si todos los patrones del conjunto fundamental tienen un 1 en la componente i.

Y eso es ni más ni menos que una operación lógica AND entre todas las componentes i; pero como la operación AND se puede caracterizar con la operación β propia de los modelos asociativos Alfa-Beta, la definición del vector de soporte $\mathbf{S}$ se hará en función de la operación β.

Definición 3.57 *Sean* $A = \{0, 1\}$, $n \in \mathbb{Z}^+$, $p \in \mathbb{Z}^+$, *con* $1 < p \leq 2^n$; *y sea* $\{(\mathbf{x}^\mu, \mathbf{x}^\mu) \mid \mu = 1, 2, ..., p\}$ *un conjunto fundamental de cardinalidad* p, *con* $\mathbf{x}^\mu \in A^n \; \forall \mu \in \{1, 2, ..., p\}$. *Se define el vector de soporte* $\mathbf{S}$, *como aquel cuyas componentes* S_i, *con* $1 \leq i \leq n$ *se calculan, a partir del conjunto fundamental, de acuerdo con la siguiente expresión:*

$$S_i = \begin{cases} \bigwedge_{k=1}^{p/2} \beta\left(x_i^{2k-1}, x_i^{2k}\right) & \text{si } p \text{ es par} \\ \beta\left[\bigwedge_{k=1}^{(p-1)/2} \beta\left(x_i^{2k-1}, x_i^{2k}\right), x_i^p\right] & \text{si } p \text{ es non} \end{cases}$$

Ejemplo 3.58 *Sean* $\mathbf{x^1}, \mathbf{x^2}, \mathbf{x^3}, \mathbf{x^4}, \mathbf{x^5}, \mathbf{x^6}, \mathbf{x^7} \in A^n$ *con* $n = 9$ *los vectores que conforman el conjunto fundamental:*

$$\mathbf{x^1}=\begin{pmatrix}0\\1\\0\\1\\0\\1\\1\\1\\1\end{pmatrix}, \mathbf{x^2}=\begin{pmatrix}0\\1\\0\\1\\1\\1\\1\\1\\1\end{pmatrix}, \mathbf{x^3}=\begin{pmatrix}0\\1\\1\\1\\1\\1\\0\\1\\1\end{pmatrix}, \mathbf{x^4}=\begin{pmatrix}1\\1\\0\\1\\0\\1\\1\\1\\1\end{pmatrix}, \mathbf{x^5}=\begin{pmatrix}1\\1\\1\\1\\0\\1\\0\\1\\1\end{pmatrix}, \mathbf{x^6}=\begin{pmatrix}0\\1\\1\\1\\0\\1\\1\\1\\1\end{pmatrix}, \mathbf{x^7}=\begin{pmatrix}0\\1\\0\\1\\1\\1\\0\\1\\1\end{pmatrix}$$

i) Usar la Definición 3.57 para construir el vector de soporte $\mathbf{S}$.

ii) Para cada $\mu \in \{1, 2, 3, 4, 5, 6, 7\}$ *realizar lo siguiente: a) Obtener* $\mathbf{x}^\mu|_{\mathbf{S}}$; *b) Obtener* $(\mathbf{x}^\mu|_{\mathbf{S}})|^{\mathbf{S}}$ *y c) Verificar que* $(\mathbf{x}^\mu|_{\mathbf{S}})|^{\mathbf{S}} = \mathbf{x}^\mu$.

RESPUESTAS:

i) Tomando la definición 3.57 para el cálculo del vector soporte y sabiendo que la cardinalidad del conjunto fundamental es impar $p = 7$ entonces usamos la siguiente expresión:

$$S_i = \beta\left[\bigwedge_{k=1}^{(p-1)/2} \beta\left(x_i^{2k-1}, x_i^{2k}\right), x_i^p\right]$$

$$S_i = \beta\left[\beta\left(x_i^1, x_i^2\right) \wedge \beta\left(x_i^3, x_i^4\right) \wedge \beta\left(x_i^5, x_i^6\right), x_i^7\right]$$

$$S_1 = \beta\left[\beta\left(0,0\right) \wedge \beta\left(0,1\right) \wedge \beta\left(1,0\right), 0\right] = \beta\left[\left(0 \wedge 0 \wedge 0\right), 0\right] = \beta\left[0,0\right] = 0$$

$$S_2 = \beta\left[\beta\left(1,1\right) \wedge \beta\left(1,1\right) \wedge \beta\left(1,1\right), 1\right] = \beta\left[\left(1 \wedge 1 \wedge 1\right), 1\right] = \beta\left[1,1\right] = 1$$

$$S_3 = \beta\left[\beta\left(0,0\right) \wedge \beta\left(1,0\right) \wedge \beta\left(1,1\right), 0\right] = \beta\left[\left(0 \wedge 0 \wedge 1\right), 0\right] = \beta\left[0,0\right] = 0$$
$$S_4 = \beta\left[\beta\left(1,1\right) \wedge \beta\left(1,1\right) \wedge \beta\left(1,1\right), 1\right] = \beta\left[\left(1 \wedge 1 \wedge 1\right), 1\right] = \beta\left[1,1\right] = 1$$
$$S_5 = \beta\left[\beta\left(0,1\right) \wedge \beta\left(1,0\right) \wedge \beta\left(0,0\right), 1\right] = \beta\left[\left(0 \wedge 0 \wedge 0\right), 1\right] = \beta\left[0,1\right] = 0$$
$$S_6 = \beta\left[\beta\left(1,1\right) \wedge \beta\left(1,1\right) \wedge \beta\left(1,1\right), 1\right] = \beta\left[\left(1 \wedge 1 \wedge 1\right), 1\right] = \beta\left[1,1\right] = 1$$
$$S_7 = \beta\left[\beta\left(1,1\right) \wedge \beta\left(0,1\right) \wedge \beta\left(0,1\right), 0\right] = \beta\left[\left(1 \wedge 0 \wedge 0\right), 0\right] = \beta\left[0,0\right] = 0$$
$$S_8 = \beta\left[\beta\left(1,1\right) \wedge \beta\left(1,1\right) \wedge \beta\left(1,1\right), 1\right] = \beta\left[\left(1 \wedge 1 \wedge 1\right), 1\right] = \beta\left[1,1\right] = 1$$
$$S_9 = \beta\left[\beta\left(1,1\right) \wedge \beta\left(1,1\right) \wedge \beta\left(1,1\right), 1\right] = \beta\left[\left(1 \wedge 1 \wedge 1\right), 1\right] = \beta\left[1,1\right] = 1$$

de ahí que el vector de soporte para este conjunto fundamental es $\mathbf{S} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \end{pmatrix}$.

ii.1-a) Para el primer vector $\mathbf{x^1} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$ la Restricción de $\mathbf{x^1}$ por $\mathbf{S}$ es $\mathbf{x^1}\big|_{\mathbf{S}} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$.

ii.1-b) De acuerdo con la definición 3.42, para obtener $\left(\mathbf{x^1}\big|_{\mathbf{S}}\right)\big|^{\mathbf{S}}$ es necesario llevar a cabo la Inserción en $\mathbf{x^1}\big|_{\mathbf{S}}$ según $\mathbf{S}$, para todas y cada una de las componentes de $\mathbf{S}$ con valor 1, es decir, la operación Inserción será efectuada $\sigma_n(\mathbf{S}) = \mathbf{5}$ veces, con lo que se obtiene la Expansión de $\mathbf{x^1}\big|_{\mathbf{S}}$ por $\mathbf{S}$: $\left(\mathbf{x^1}\big|_{\mathbf{S}}\right)\big|^{\mathbf{S}} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$.

ii.1-c) Al comparar el vector obtenido mediante la Expansión de $\mathbf{x^1}\big|_{\mathbf{S}}$ por $\mathbf{S}$ contra el vector original $\mathbf{x^1}$, se puede observar que todas las posiciones son correctamente

recuperadas:

$$\mathbf{x^1} = \begin{pmatrix}0\\1\\0\\1\\0\\1\\1\\1\\1\end{pmatrix} \qquad \left(\mathbf{x^1}\big|_{\mathbf{S}}\right)\Big|^{\mathbf{S}} = \begin{pmatrix}0\\1\\0\\1\\0\\1\\1\\1\\1\end{pmatrix}$$

ii.2-a) Para el segundo vector $\mathbf{x^2}= \begin{pmatrix}0\\1\\0\\1\\1\\1\\1\\1\\1\end{pmatrix}$ la Restricción de $\mathbf{x^2}$ por $\mathbf{S}$ es $\mathbf{x^2}\big|_{\mathbf{S}} = \begin{pmatrix}0\\0\\1\\1\end{pmatrix}$.

ii.2-b) De acuerdo con la definición 3.42, para obtener $\left(\mathbf{x^2}\big|_{\mathbf{S}}\right)\Big|^{\mathbf{S}}$ es necesario llevar a cabo la Inserción en $\mathbf{x^2}\big|_{\mathbf{S}}$ según $\mathbf{S}$, para todas y cada una de las componentes de $\mathbf{S}$ con valor 1, es decir, la operación Inserción será efectuada $\sigma_n(\mathbf{S}) = \mathbf{5}$ veces, con lo que se obtiene la Expansión de $\mathbf{x^2}\big|_{\mathbf{S}}$ por $\mathbf{S}$: $\left(\mathbf{x^2}\big|_{\mathbf{S}}\right)\Big|^{\mathbf{S}} = \begin{pmatrix}0\\1\\0\\1\\1\\1\\1\\1\\1\end{pmatrix}$.

ii.2-c) Al comparar el vector obtenido mediante la Expansión de $\mathbf{x^2}\big|_{\mathbf{S}}$ por $\mathbf{S}$ contra el vector original $\mathbf{x^2}$, se puede observar que todas las posiciones son correctamente recuperadas:

$$\mathbf{x^2} = \begin{pmatrix}0\\1\\0\\1\\1\\1\\1\\1\\1\end{pmatrix} \qquad \left(\mathbf{x^2}\big|_{\mathbf{S}}\right)\Big|^{\mathbf{S}} = \begin{pmatrix}0\\1\\0\\1\\1\\1\\1\\1\\1\end{pmatrix}$$

ii.3-a) Para el tercer vector $\mathbf{x^3}=\begin{pmatrix}0\\1\\1\\1\\1\\1\\0\\1\\1\end{pmatrix}$ la Restricción de $\mathbf{x^3}$ por $\mathbf{S}$ es $\mathbf{x^3}\big|_{\mathbf{S}}=\begin{pmatrix}0\\1\\1\\0\end{pmatrix}$.

ii.3-b) De acuerdo con la definición 3.42, para obtener $\left(\mathbf{x^3}\big|_{\mathbf{S}}\right)\big|^{\mathbf{S}}$ es necesario llevar a cabo la Inserción en $\mathbf{x^3}\big|_{\mathbf{S}}$ según $\mathbf{S}$, para todas y cada una de las componentes de $\mathbf{S}$ con valor 1, es decir, la operación Inserción será efectuada $\sigma_n(\mathbf{S})=\mathbf{5}$ veces, con lo que se obtiene la Expansión de $\mathbf{x^3}\big|_{\mathbf{S}}$ por $\mathbf{S}$: $\left(\mathbf{x^3}\big|_{\mathbf{S}}\right)\big|^{\mathbf{S}}=\begin{pmatrix}0\\1\\1\\1\\1\\1\\0\\1\\1\end{pmatrix}$.

ii.3-c) Al comparar el vector obtenido mediante la Expansión de $\mathbf{x^3}\big|_{\mathbf{S}}$ por $\mathbf{S}$ contra el vector original $\mathbf{x^3}$, se puede observar que todas las posiciones son correctamente recuperadas:

$$\mathbf{x^3}=\begin{pmatrix}0\\1\\1\\1\\1\\1\\0\\1\\1\end{pmatrix} \qquad \left(\mathbf{x^3}\big|_{\mathbf{S}}\right)\big|^{\mathbf{S}}=\begin{pmatrix}0\\1\\1\\1\\1\\1\\0\\1\\1\end{pmatrix}$$

ii.4-a) Para el cuarto vector $\mathbf{x^4}=\begin{pmatrix}1\\1\\0\\1\\0\\1\\1\\1\\1\end{pmatrix}$ la Restricción de $\mathbf{x^4}$ por $\mathbf{S}$ es $\mathbf{x^4}\big|_{\mathbf{S}}=\begin{pmatrix}1\\0\\0\\1\end{pmatrix}$.

ii.4-b) De acuerdo con la definición 3.42, para obtener $\left(\mathbf{x^4}\big|_{\mathbf{S}}\right)\big|^{\mathbf{S}}$ es necesario llevar a cabo la Inserción en $\mathbf{x^4}\big|_{\mathbf{S}}$ según $\mathbf{S}$, para todas y cada una de las componentes de $\mathbf{S}$ con valor 1, es decir, la operación Inserción será efectuada $\sigma_n(\mathbf{S})=\mathbf{5}$ veces, con lo que

se obtiene la Expansión de $\mathbf{x^4}\big|_{\mathbf{S}}$ por $\mathbf{S}$: $\left(\mathbf{x^4}\big|_{\mathbf{S}}\right)\big|^{\mathbf{S}} = \begin{pmatrix}1\\1\\0\\1\\0\\1\\1\\1\\1\end{pmatrix}$.

ii.4-c) Al comparar el vector obtenido mediante la Expansión de $\mathbf{x^4}\big|_{\mathbf{S}}$ por $\mathbf{S}$ contra el vector original $\mathbf{x^4}$, se puede observar que todas las posiciones son correctamente recuperadas:

$$\mathbf{x^4} = \begin{pmatrix}1\\1\\0\\1\\0\\1\\1\\1\\1\end{pmatrix} \qquad \left(\mathbf{x^4}\big|_{\mathbf{S}}\right)\big|^{\mathbf{S}} = \begin{pmatrix}1\\1\\0\\1\\0\\1\\1\\1\\1\end{pmatrix}$$

ii.5-a) Para el quinto vector $\mathbf{x^5}= \begin{pmatrix}1\\1\\1\\1\\0\\1\\0\\1\\1\end{pmatrix}$ la Restricción de $\mathbf{x^5}$ por $\mathbf{S}$ es $\mathbf{x^5}\big|_{\mathbf{S}} = \begin{pmatrix}1\\1\\0\\0\end{pmatrix}$.

ii.5-b) De acuerdo con la definición 3.42, para obtener $\left(\mathbf{x^5}\big|_{\mathbf{S}}\right)\big|^{\mathbf{S}}$ es necesario llevar a cabo la Inserción en $\mathbf{x^5}\big|_{\mathbf{S}}$ según $\mathbf{S}$, para todas y cada una de las componentes de $\mathbf{S}$ con valor 1, es decir, la operación Inserción será efectuada $\sigma_n(\mathbf{S}) = \mathbf{5}$ veces, con lo que se obtiene la Expansión de $\mathbf{x^5}\big|_{\mathbf{S}}$ por $\mathbf{S}$: $\left(\mathbf{x^5}\big|_{\mathbf{S}}\right)\big|^{\mathbf{S}} = \begin{pmatrix}1\\1\\1\\1\\0\\1\\0\\1\\1\end{pmatrix}$.

ii.5-c) Al comparar el vector obtenido mediante la Expansión de $\mathbf{x^5}\big|_{\mathbf{S}}$ por $\mathbf{S}$ contra el vector original $\mathbf{x^5}$, se puede observar que todas las posiciones son correctamente

recuperadas:

$$\mathbf{x^5} = \begin{pmatrix}1\\1\\1\\1\\0\\1\\0\\1\\1\end{pmatrix} \qquad \left(\mathbf{x^5}\big|_{\mathbf{S}}\right)\Big|^{\mathbf{S}} = \begin{pmatrix}1\\1\\1\\1\\0\\1\\0\\1\\1\end{pmatrix}$$

ii.6-a) Para el sexto vector $\mathbf{x^6}= \begin{pmatrix}0\\1\\1\\1\\0\\1\\1\\1\\1\end{pmatrix}$ la Restricción de $\mathbf{x^6}$ por $\mathbf{S}$ es $\mathbf{x^6}\big|_{\mathbf{S}} = \begin{pmatrix}0\\1\\0\\1\end{pmatrix}$.

ii.6-b) De acuerdo con la definición 3.42, para obtener $\left(\mathbf{x^6}\big|_{\mathbf{S}}\right)\Big|^{\mathbf{S}}$ es necesario llevar a cabo la Inserción en $\mathbf{x^6}\big|_{\mathbf{S}}$ según $\mathbf{S}$, para todas y cada una de las componentes de $\mathbf{S}$ con valor 1, es decir, la operación Inserción será efectuada $\sigma_n(\mathbf{S}) = \mathbf{5}$ veces, con lo que se obtiene la Expansión de $\mathbf{x^6}\big|_{\mathbf{S}}$ por $\mathbf{S}$: $\left(\mathbf{x^6}\big|_{\mathbf{S}}\right)\Big|^{\mathbf{S}} = \begin{pmatrix}0\\1\\1\\1\\0\\1\\1\\1\\1\end{pmatrix}$.

ii.6-c) Al comparar el vector obtenido mediante la Expansión de $\mathbf{x^6}\big|_{\mathbf{S}}$ por $\mathbf{S}$ contra el vector original $\mathbf{x^6}$, se puede observar que todas las posiciones son correctamente recuperadas:

$$\mathbf{x^6} = \begin{pmatrix}0\\1\\1\\1\\0\\1\\1\\1\\1\end{pmatrix} \qquad \left(\mathbf{x^6}\big|_{\mathbf{S}}\right)\Big|^{\mathbf{S}} = \begin{pmatrix}0\\1\\1\\1\\0\\1\\1\\1\\1\end{pmatrix}$$

ii.7-a) Para el séptimo vector $\mathbf{x^7}=\begin{pmatrix}0\\1\\0\\1\\1\\1\\0\\1\\1\end{pmatrix}$ la Restricción de $\mathbf{x^7}$ por $\mathbf{S}$ es $\mathbf{x^7}\big|_{\mathbf{S}}=\begin{pmatrix}0\\0\\1\\0\end{pmatrix}$.

ii.7-b) De acuerdo con la definición 3.42, para obtener $\left(\mathbf{x^7}\big|_{\mathbf{S}}\right)\big|^{\mathbf{S}}$ es necesario llevar a cabo la Inserción en $\mathbf{x^7}\big|_{\mathbf{S}}$ según $\mathbf{S}$, para todas y cada una de las componentes de $\mathbf{S}$ con valor 1, es decir, la operación Inserción será efectuada $\sigma_n(\mathbf{S})=\mathbf{5}$ veces, con lo que se obtiene la Expansión de $\mathbf{x^7}\big|_{\mathbf{S}}$ por $\mathbf{S}$: $\left(\mathbf{x^7}\big|_{\mathbf{S}}\right)\big|^{\mathbf{S}}=\begin{pmatrix}0\\1\\0\\1\\1\\1\\0\\1\\1\end{pmatrix}$.

ii.7-c) Al comparar el vector obtenido mediante la Expansión de $\mathbf{x^7}\big|_{\mathbf{S}}$ por $\mathbf{S}$ contra el vector original $\mathbf{x^7}$, se puede observar que todas las posiciones son correctamente recuperadas:

$$\mathbf{x^7}=\begin{pmatrix}0\\1\\0\\1\\1\\1\\0\\1\\1\end{pmatrix} \qquad \left(\mathbf{x^7}\big|_{\mathbf{S}}\right)\big|^{\mathbf{S}}=\begin{pmatrix}0\\1\\0\\1\\1\\1\\0\\1\\1\end{pmatrix}$$

Ejemplo 3.59 *Sean* $\mathbf{x^1},\mathbf{x^2},\mathbf{x^3}\in A^n$ *con* $n=4$ *los vectores que conforman el conjunto fundamental:*

$$\mathbf{x^1}=\begin{pmatrix}0\\1\\1\\1\end{pmatrix},\mathbf{x^2}=\begin{pmatrix}1\\1\\0\\1\end{pmatrix},\mathbf{x^3}=\begin{pmatrix}0\\0\\1\\0\end{pmatrix}$$

Usar la Definición 3.57 para construir el vector de soporte $\mathbf{S}$.

Tomando la definición 3.57 para el cálculo del vector soporte y sabiendo que la cardinalidad del conjunto fundamental es impar $p=3$ *entonces usamos la siguiente expresión:*

$$S_i = \beta \left[\bigwedge_{k=1}^{(p-1)/2} \beta \left(x_i^{2k-1}, x_i^{2k} \right), x_i^p \right]$$

$$S_i = \beta \left[\beta \left(x_i^1, x_i^2 \right), x_i^3 \right]$$

$$S_1 = \beta \left[\beta \left(0, 1 \right), 0 \right] = \beta \left[0, 0 \right] = 0$$

$$S_2 = \beta \left[\beta \left(1, 1 \right), 0 \right] = \beta \left[1, 0 \right] = 0$$

$$S_3 = \beta \left[\beta \left(1, 0 \right), 1 \right] = \beta \left[0, 0 \right] = 0$$

$$S_4 = \beta \left[\beta \left(1, 1 \right), 0 \right] = \beta \left[1, 0 \right] = 0$$

de ahí que el vector de soporte para este conjunto fundamental es $\mathbf{S} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$

Aparece de pronto un problema: un vector de soporte $\mathbf{S}$ que tiene todas sus componentes con valor 0; es decir, $\mathbf{S} = \mathbf{0}$. Pero en las principales definiciones y resultados previos se pide que el vector $\mathbf{x} \neq \mathbf{0}$, y en este caso debería cumplirse que $\mathbf{S} \neq \mathbf{0}$ para poder aplicar las definiciones y resultados previos. La razón por la que se pone como condición que $\mathbf{x} \neq \mathbf{0}$ es para evitar que se pesenten casos anómalos, pero el vector de soporte $\mathbf{S} = \mathbf{0}$ surgió de un ejemplo real de conjunto fundamental; ¿qué se puede hacer?.

Reflexionemos un poco: en el ejemplo 3.59, ¿tiene algún significado la expresión $\mathbf{x^1}\big|_{\mathbf{S}}$ si sabemos que $\mathbf{S} = \mathbf{0}$? Veamos: para obtener $\mathbf{x^1}\big|_{\mathbf{S}}$, de acuerdo con la Definición 3.30 deberíamos de aplicar iteradamente la operación Eliminación en $\mathbf{x}^1$ según $\mathbf{S}$, para todas y cada una de las componentes de $\mathbf{S}$ con valor 1, pero...¡$\mathbf{S}$ no tiene componentes con valor 1!; es decir, deberíamos de aplicar la operación Eliminación en $\mathbf{x^1}$ según $\mathbf{S}$ cero veces, y eso significa que se eliminan cero componentes del vector $\mathbf{x^1}$, lo que significa, a su vez, que el vector $\mathbf{x^1}$ no sufre modificación alguna; queda igual.

Lo anterior nos sugiere que $\mathbf{x^1}\big|_{\mathbf{0}} = \mathbf{x^1}$. No debe extrañarnos una definición semejante, pues en las matemáticas se presenta una situación similar con la definición del factorial de un número; según la referencia [102], el factorial de un número entero positivo n, denotado por $n!$, se define como $n! = n \cdot (n-1) \cdot (n-2) \cdot ... \cdot 3 \cdot 2 \cdot 1$, pero ¿acaso tiene sentido la expresión 0!? La respuesta es sí, y por definición, se dice que $0! = 1$. Tomando como inspiración esta definición del factorial de cero, se plantea la siguiente definición.

Definición 3.60 *Sea* $A = \{0, 1\}$*, y sea* $\mathbf{x}$ *un vector columna* $\mathbf{x} \in A^n$*, con* $n \in \mathbb{Z}^+$*. Se define la Restricción de* $\mathbf{x}$ *por* $\mathbf{0}$ *como:* $\mathbf{x}|_{\mathbf{0}} = \mathbf{x}$*; asimismo, se define la Expansión de* $\mathbf{x}$ *por* $\mathbf{0}$ *como:* $\mathbf{x}|^{\mathbf{0}} = \mathbf{x}$*.*

Ejemplo 3.61 *En referencia al Ejemplo 3.59, para cada* $\mu \in \{1, 2, 3\}$ *obtener* $(\mathbf{x}^\mu|_{\mathbf{S}})|^{\mathbf{S}}$*.*

Tomemos arbitrariamente un valor de $\mu \in \{1, 2, 3\}$. Dado que $\mathbf{S} = \mathbf{0}$, usemos la Definición 3.60, para obtener: $(\mathbf{x}^\mu|_{\mathbf{S}})|^{\mathbf{S}} = \mathbf{x}^\mu|^{\mathbf{S}} = \mathbf{x}^\mu$. Como el valor de μ se escogió de manera arbitraria, lo anterior se cumple para todas las $\mu \in \{1, 2, 3\}$.

Teorema 3.62 *Sean* $A = \{0, 1\}$, $n \in \mathbb{Z}^+$, $p \in \mathbb{Z}^+$, *con* $1 < p \leq 2^n$; *y sea* $\{(\mathbf{x}^\mu, \mathbf{x}^\mu) \mid \mu = 1, 2, ..., p\}$ *un conjunto fundamental de cardinalidad* p, *con* $\mathbf{x}^\mu \in A^n \; \forall \mu \in \{1, 2, ..., p\}$ *cuyo vector de soporte es* $\mathbf{S}$, *de acuerdo con la Definición 3.57. La siguiente expresión se cumple para* $\forall \mu \in \{1, 2, ..., p\}$:

$$(\mathbf{x}^\mu|_{\mathbf{S}})|^{\mathbf{S}} = \mathbf{x}^\mu$$

Demostración.- De acuerdo con la Definición 3.57, dado un valor $\mu \in \{1, 2, ..., p\}$ y un valor $i \in \{1, 2, ..., n\}$, no es posible que ocurra $x_i^\mu = 1$ y $S_i = 0$. Si identificamos al vector $\mathbf{x}^\mu$ con el vector $\mathbf{x}$ y al vector $\mathbf{S}$ con el vector $\mathbf{y}$ del Teorema 3.56, el CASO 4 nunca ocurre; es decir, se cumple que $\left[(\mathbf{x}^\mu|_{\mathbf{S}})|^{\mathbf{S}}\right]_i = x_i^\mu$, $\forall \mu \in \{1, 2, ..., p\}$, $\forall i \in \{1, 2, ..., n\}$. Por lo tanto, La siguiente expresión se cumple para $\forall \mu \in \{1, 2, ..., p\}$: $(\mathbf{x}^\mu|_{\mathbf{S}})|^{\mathbf{S}} = \mathbf{x}^\mu$.■

3.4. Dos nuevas transformadas

El Teorema 3.62 es de suma importancia para el nuevo modelo de máquinas asociativas Alfa-Beta con soporte vectorial, dado que garantiza la recuperación del conjunto fundamental en su totalidad.

Recuperar todo el conjunto fundamental, sin falla, es uno de los requisitos básicos que debe cumplir un reconocedor de patrones; pero eso no es todo: la relevancia del nuevo modelo de máquinas asociativas Alfa-Beta con soporte vectorial se pondrá de manifiesto cuando se exhiba que este modelo es robusto ante alteraciones aditivas, sustractivas y, sobre todo, alteraciones mezcladas.

Los experimentos muestran sin duda que, en efecto, las máquinas asociativas Alfa-Beta con soporte vectorial son robustas ante estos tres tipos de alteraciones, a diferencia de otros modelos que fallaban cuando a la entrada había un patrón con alteraciones mezcladas respecto a uno de los patrones del conjunto fundamental.

A fin de estar en posibilidades de formular el algoritmo principal de este trabajo de tesis, se requiere la definición de dos transformadas originales: la Transformada $\boldsymbol{\tau}$ y la Transformada θ.

Definición 3.63 *Sea* $A = \{0, 1\}$, *y sean* $\mathbf{x}$ *y* $\mathbf{y}$ *dos vectores* $\mathbf{x} \in A^n$, $\mathbf{y} \in A^n$, *con* $n \in \mathbb{Z}^+$. *La Transformada* $\boldsymbol{\tau}$ *de* $\mathbf{x}$ *con respecto a* $\mathbf{y}$ *da como resultado un vector* $\boldsymbol{\tau}(\mathbf{x}, \mathbf{y})$ *de dimensión* n, *cuya* $i-$ *ésima componente se calcula como sigue:*

$$[\boldsymbol{\tau}(\mathbf{x}, \mathbf{y})]_i = \beta[x_i, \alpha(0, y_i)]$$

Ejemplo 3.64 *Sean* $\mathbf{x} = \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$ *y* $\mathbf{y} = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \end{pmatrix}$; *obtener* $\boldsymbol{\tau}(\mathbf{x}, \mathbf{y})$, *la Transformada* $\boldsymbol{\tau}$ *de* $\mathbf{x}$ *con respecto a* $\mathbf{y}$.

De acuerdo con la definición 3.63, el resultado de la Transformada $\boldsymbol{\tau}$ de $\mathbf{x}$ con respecto a $\mathbf{y}$ es un vector de dimensión n, cuya $i-$ ésima componente se calcula haciendo uso de la expresión $[\boldsymbol{\tau}(\mathbf{x},\mathbf{y})]_i = \beta[x_i, \alpha(0, y_i)]$; apliquemos esta expresión en cada una de las 8 componentes:

$$[\boldsymbol{\tau}(\mathbf{x},\mathbf{y})]_1 = \beta[x_1, \alpha(0, y_1)] = \beta[0, \alpha(0,1)] = \beta[0,0] = 0$$

$$[\boldsymbol{\tau}(\mathbf{x},\mathbf{y})]_2 = \beta[x_2, \alpha(0, y_2)] = \beta[1, \alpha(0,0)] = \beta[1,1] = 1$$

$$[\boldsymbol{\tau}(\mathbf{x},\mathbf{y})]_3 = \beta[x_3, \alpha(0, y_3)] = \beta[1, \alpha(0,1)] = \beta[1,0] = 0$$

$$[\boldsymbol{\tau}(\mathbf{x},\mathbf{y})]_4 = \beta[x_4, \alpha(0, y_4)] = \beta[0, \alpha(0,1)] = \beta[0,0] = 0$$

$$[\boldsymbol{\tau}(\mathbf{x},\mathbf{y})]_5 = \beta[x_5, \alpha(0, y_5)] = \beta[0, \alpha(0,0)] = \beta[0,1] = 0$$

$$[\boldsymbol{\tau}(\mathbf{x},\mathbf{y})]_6 = \beta[x_6, \alpha(0, y_6)] = \beta[1, \alpha(0,0)] = \beta[1,1] = 1$$

$$[\boldsymbol{\tau}(\mathbf{x},\mathbf{y})]_7 = \beta[x_7, \alpha(0, y_7)] = \beta[1, \alpha(0,1)] = \beta[1,0] = 0$$

$$[\boldsymbol{\tau}(\mathbf{x},\mathbf{y})]_8 = \beta[x_8, \alpha(0, y_8)] = \beta[0, \alpha(0,1)] = \beta[0,0] = 0$$

Así que la Transformada $\boldsymbol{\tau}$ de $\mathbf{x}$ con respecto a $\mathbf{y}$ es:

$$\boldsymbol{\tau}(\mathbf{x},\mathbf{y}) = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}$$

Se puede observar que el vector obtenido $\boldsymbol{\tau}(\mathbf{x},\mathbf{y})$, tiene valor 1 en aquellas posiciones donde existen alteraciones sustractivas del vector $\mathbf{y}$ con respecto al vector $\mathbf{x}$.

Nota 3.65 *Una alteración sustractiva del vector* $\mathbf{y}$ *con respecto al vector* $\mathbf{x}$*, puede considerarse también como una alteración aditiva del vector* $\mathbf{x}$ *con respecto a* $\mathbf{y}$*.*

Ejemplo 3.66 *Sean* $\mathbf{x}$ *y* $\mathbf{y}$ *los mismos vectores del Ejemplo 3.64; obtener* $\boldsymbol{\tau}(\mathbf{y},\mathbf{x})$*, la Transformada* $\boldsymbol{\tau}$ *de* $\mathbf{y}$ *con respecto a* $\mathbf{x}$*.*

De acuerdo con la definición 3.63, el resultado de la Transformada $\boldsymbol{\tau}$ de $\mathbf{y}$ con respecto a $\mathbf{x}$ es un vector de dimensión n, cuya $i-$ ésima componente se calcula haciendo uso de la expresión $[\boldsymbol{\tau}(\mathbf{y},\mathbf{x})]_i = \beta[y_i, \alpha(0, x_i)]$; apliquemos esta expresión en cada una de las 8 componentes:

$$[\boldsymbol{\tau}(\mathbf{y},\mathbf{x})]_1 = \beta[y_1, \alpha(0, x_1)] = \beta[1, \alpha(0,0)] = \beta[1,1] = 1$$

$$[\boldsymbol{\tau}(\mathbf{y},\mathbf{x})]_2 = \beta[y_2, \alpha(0, x_2)] = \beta[0, \alpha(0,1)] = \beta[0,0] = 0$$

$$[\boldsymbol{\tau}(\mathbf{y},\mathbf{x})]_3 = \beta[y_3, \alpha(0, x_3)] = \beta[1, \alpha(0,1)] = \beta[1,0] = 0$$

$$[\boldsymbol{\tau}(\mathbf{y},\mathbf{x})]_4 = \beta[y_4, \alpha(0, x_4)] = \beta[1, \alpha(0,0)] = \beta[1,1] = 1$$

$$[\boldsymbol{\tau}(\mathbf{y},\mathbf{x})]_5 = \beta[y_5, \alpha(0, x_5)] = \beta[0, \alpha(0,0)] = \beta[0,1] = 0$$
$$[\boldsymbol{\tau}(\mathbf{y},\mathbf{x})]_6 = \beta[y_6, \alpha(0, x_6)] = \beta[0, \alpha(0,1)] = \beta[0,0] = 0$$
$$[\boldsymbol{\tau}(\mathbf{y},\mathbf{x})]_7 = \beta[y_7, \alpha(0, x_7)] = \beta[1, \alpha(0,1)] = \beta[1,0] = 0$$
$$[\boldsymbol{\tau}(\mathbf{y},\mathbf{x})]_8 = \beta[y_8, \alpha(0, x_8)] = \beta[1, \alpha(0,0)] = \beta[1,1] = 1$$

Así que la Transformada $\boldsymbol{\tau}$ de $\mathbf{y}$ con respecto a $\mathbf{x}$ es:

$$\boldsymbol{\tau}(\mathbf{y},\mathbf{x}) = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

Se puede observar que el vector obtenido $\boldsymbol{\tau}(\mathbf{y},\mathbf{x})$, tiene valor 1 en aquellas posiciones donde existen alteraciones sustractivas del vector $\mathbf{x}$ con respecto a $\mathbf{y}$; o de manera equivalente, donde existen

alteraciones aditivas del vector $\mathbf{y}$ con respecto al vector $\mathbf{x}$.

Definición 3.67 *Sea $A = \{0,1\}$, y sean $\mathbf{x}$ y $\mathbf{y}$ dos vectores $\mathbf{x} \in A^n$, $\mathbf{y} \in A^n$, con $n \in \mathbb{Z}^+$. La Transformada θ de $\mathbf{x}$ con respecto a $\mathbf{y}$ es un escalar $\theta(\mathbf{x},\mathbf{y})$ definido como sigue:*

$$\theta(\mathbf{x},\mathbf{y}) = \sigma_n[\boldsymbol{\tau}(\mathbf{x},\mathbf{y})] + \sigma_n[\boldsymbol{\tau}(\mathbf{y},\mathbf{x})]$$

Ejemplo 3.68 *Sean $\mathbf{x}$ y $\mathbf{y}$ los mismos vectores del Ejemplo 3.64; obtener $\theta(\mathbf{x},\mathbf{y})$, la Transformada θ de $\mathbf{x}$ con respecto a $\mathbf{y}$.*

De acuerdo con el Ejemplo 3.64, la Transformada $\boldsymbol{\tau}$ de $\mathbf{x}$ con respecto a $\mathbf{y}$ es el vector siguiente:

$$\boldsymbol{\tau}(\mathbf{x},\mathbf{y}) = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}$$

Al aplicar $\sigma_n[\boldsymbol{\tau}(\mathbf{x},\mathbf{y})]$ de acuerdo con la Definición 3.8, obtenemos el escalar $\sigma_8[\boldsymbol{\tau}(\mathbf{x},\mathbf{y})] = 2$.

Del mismo modo, usando el resultado del Ejemplo 3.66, la Transformada $\boldsymbol{\tau}$ de $\mathbf{y}$ con respecto a $\mathbf{x}$:

$$\boldsymbol{\tau}(\mathbf{y},\mathbf{x}) = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

Al aplicar $\sigma_n[\boldsymbol{\tau}(\mathbf{x},\mathbf{y})]$ de acuerdo con la Definición 3.8, obtenemos el escalar $\sigma_8[\boldsymbol{\tau}(\mathbf{y},\mathbf{x})] = 3$.

Una vez que se tienen calculados los valores de $\sigma_8[\boldsymbol{\tau}(\mathbf{x},\mathbf{y})]$ y $\sigma_8[\boldsymbol{\tau}(\mathbf{y},\mathbf{x})]$ es posible obtener la Transformada θ de $\mathbf{x}$ con respecto a $\mathbf{y}$ usando la definición 3.67.

$$\theta(\mathbf{x},\mathbf{y}) = 5$$

3.5. Algoritmo Principal

Se asume que se tiene un problema de reconocimiento de patrones, donde el conjunto fundamental es de la forma $\{(\mathbf{x}^\mu,\mathbf{x}^\mu) \mid \mu = 1,2,...,p\}$, con $\mathbf{x}^\mu \in A^n\ \forall \mu \in \{1,2,...,p\}$, siendo $n,p \in \mathbb{Z}^+$ y $A = \{0,1\}$.

Fase de aprendizaje del nuevo modelo

1. A partir del conjunto fundamental, calcular el vector de soporte $\mathbf{S}$, de acuerdo con la Definición 3.57.

2. Para cada $\mu \in \{1,2,...,p\}$, obtener $\mathbf{x}^\mu|_{\mathbf{S}}$ de acuerdo con las Definiciones 3.30 y 3.60. A partir de estos resultados se forma el conjunto fudamental restringido $\{(\mathbf{x}^\mu|_{\mathbf{S}}, \mathbf{x}^\mu|_{\mathbf{S}}) \,|\mu = 1,2,\ldots,p\}$.

3. Para cada $\mu \in \{1,2,...,p\}$, obtener $\overline{\mathbf{x}^\mu}$, el vector negado de $\mathbf{x}^\mu$, usando la Definición 3.6. Con los p vectores negados, se forma el conjunto fundamental negado $\{(\overline{\mathbf{x}^\mu},\overline{\mathbf{x}^\mu}) \mid \mu = 1,2,...,p\}$.

4. A partir del conjunto fundamental negado, calcular el vector de soporte $\widehat{\mathbf{S}}$, de acuerdo con la Definición 3.57.

5. Para cada $\mu \in \{1,2,...,p\}$, obtener $\overline{\mathbf{x}^\mu}|_{\widehat{\mathbf{S}}}$ de acuerdo con las Definiciones 3.30 y 3.60. A partir de estos resultados se forma el conjunto fudamental negado restringido $\left\{\left(\overline{\mathbf{x}^\mu}|_{\widehat{\mathbf{S}}}, \overline{\mathbf{x}^\mu}|_{\widehat{\mathbf{S}}}\right) |\mu = 1,2,\ldots,p\right\}$.

Fase de recuperación del nuevo modelo

Sea $\tilde{\mathbf{x}} \in A^n$ un patrón de entrada cuyo patrón asociado $\mathbf{x}^\mu$ se desconoce de antemano.

1. Obtener la Restricción $\tilde{\mathbf{x}}|_{\mathbf{S}}$ de acuerdo con las Definiciones 3.30 y 3.60.

2. Para cada $\mu \in \{1, 2, ..., p\}$, obtener $\boldsymbol{\tau}\left(\tilde{\mathbf{x}}|_{\mathbf{S}}, \mathbf{x}^{\mu}|_{\mathbf{S}}\right)$, de acuerdo con la Definición 3.63.

3. Para cada $\mu \in \{1, 2, ..., p\}$, obtener, $\boldsymbol{\tau}\left(\mathbf{x}^{\mu}|_{\mathbf{S}}, \tilde{\mathbf{x}}|_{\mathbf{S}}\right)$, de acuerdo con la definición 3.63.

4. Para cada $\mu \in \{1, 2, ..., p\}$, obtener $\theta\left(\tilde{\mathbf{x}}|_{\mathbf{S}}, \mathbf{x}^{\mu}|_{\mathbf{S}}\right)$, de acuerdo con la definición 3.67.

5. Encontrar $\psi \in \{1, 2, ..., p\}$ tal que $\theta\left(\tilde{\mathbf{x}}|_{\mathbf{S}}, \mathbf{x}^{\psi}\big|_{\mathbf{S}}\right) = \bigwedge_{\mu=1}^{p} \theta\left(\tilde{\mathbf{x}}|_{\mathbf{S}}, \mathbf{x}^{\mu}|_{\mathbf{S}}\right)$.

6. Obtener $\overline{\tilde{\mathbf{x}}}$, el vector negado de $\tilde{\mathbf{x}}$, usando la Definición 3.6.

7. Obtener la Restricción $\overline{\tilde{\mathbf{x}}}|_{\widehat{\mathbf{S}}}$ de acuerdo con las Definiciones 3.30 y 3.60.

8. Para cada $\mu \in \{1, 2, ..., p\}$, obtener $\boldsymbol{\tau}\left(\overline{\tilde{\mathbf{x}}}|_{\widehat{\mathbf{S}}}, \overline{\mathbf{x}^{\mu}}|_{\widehat{\mathbf{S}}}\right)$, de acuerdo con la Definición 3.63.

9. Para cada $\mu \in \{1, 2, ..., p\}$, obtener, $\boldsymbol{\tau}\left(\overline{\mathbf{x}^{\mu}}|_{\widehat{\mathbf{S}}}, \overline{\tilde{\mathbf{x}}}|_{\widehat{\mathbf{S}}}\right)$, de acuerdo con la definición 3.63.

10. Para cada $\mu \in \{1, 2, ..., p\}$, obtener $\theta\left(\overline{\tilde{\mathbf{x}}}|_{\widehat{\mathbf{S}}}, \overline{\mathbf{x}^{\mu}}|_{\widehat{\mathbf{S}}}\right)$, de acuerdo con la definición 3.67.

11. Encontrar $\varphi \in \{1, 2, ..., p\}$ tal que $\theta\left(\overline{\tilde{\mathbf{x}}}|_{\widehat{\mathbf{S}}}, \overline{\mathbf{x}^{\varphi}}|_{\widehat{\mathbf{S}}}\right) = \bigwedge_{\mu=1}^{p} \theta\left(\overline{\tilde{\mathbf{x}}}|_{\widehat{\mathbf{S}}}, \overline{\mathbf{x}^{\mu}}|_{\widehat{\mathbf{S}}}\right)$.

12. Si $\theta\left(\tilde{\mathbf{x}}|_{\mathbf{S}}, \mathbf{x}^{\psi}\big|_{\mathbf{S}}\right) \leq \theta\left(\overline{\tilde{\mathbf{x}}}|_{\widehat{\mathbf{S}}}, \overline{\mathbf{x}^{\varphi}}|_{\widehat{\mathbf{S}}}\right)$, realizar la asignación $\omega = \psi$; de otro modo, realizar la asignación $\omega = \varphi$.

13. Obtener $\left(\mathbf{x}^{\omega}|_{\mathbf{S}}\right)|^{\mathbf{S}}$, que por el Teorema 3.62 es precisamente el vector $\mathbf{x}^{\omega}$.

Nota 3.69 *Si $\tilde{\mathbf{x}}$ es uno de los patrones del conjunto fundamental, el Teorema 3.62 garantiza que la recuperación es correcta.*

En el Anexo A se muesta el diagrama de flujo del algoritmo de las máquinas asociativas Alfa-Beta con soporte vectorial, en ambas fases: de aprendizaje y de recuperación

3.6. Complejidad del algoritmo propuesto

Una medida de la eficiencia de un algoritmo es el tiempo usado por la computadora para resolver un problema utilizando dicho algoritmo. El análisis del tiempo requerido para resolver un problema de un tamaño en particular es una medida de la complejidad del algoritmo utilizado para resolver ese problema. La complejidad en tiempo de un algoritmo se puede expresar en términos del número de operaciones usadas por el algoritmo cuando la entrada tiene un tamaño n. Para el análisis del costo temporal se utiliza un modelo formal de máquina y el conjunto de instrucciones asociados a ese modelo. Aquí utilizamos la RAM (Random Access Memory) [100] caracterizada por

un conjunto de instrucciones limitada que se ejecutan en la unidad de tiempo. Por lo tanto, el costo temporal del algoritmo puede establecerse contando la cantidad de instrucciones requeridas para su implementación.

De lo anterior se tienen dos observaciones:

- Todas las operaciones lógicas y de comparación tienen un costo unitario
- Dado un conjunto de instrucciones para un cálculo, el costo total de un paso determinado es el producto de ese costo veces la cantidad de instrucciones requerida (**for**).

Entonces, es posible analizar la complejidad en tiempo del algoritmo propuesto al estudiar cuántas operaciones elementales necesita realizar. Tomando como base los diagramas de flujo mostrados en el Anexo A y los primeros 5 pasos del algoritmo en pseudocódigo como una muestra, se realizó el análisis de complejidad del algoritmo. Las restantes 13 líneas del algoritmo fueron analizadas del propio código.

Al final de cada paso del algoritmo en pseudocódigo, se presenta una breve explicación del grado de dificultad.

PASO 1 (Vector soporte)

```
//svmab.h
bool **xu;
bool **nxu;
void paso1(void)
 {
  int i,k,min;
  for(i=0;i<n;i++)
   {
    min=1;
    if(p%2==0)
     {
      for(k=0;k<p/2;k++)
       {
        if(Beta[x[2*k-1][i],x[2*k][i]==0)
         {
          min=0;
          break;
         }
       }
      S[i]=min;
     }
    else
     {
      for(k=0;k<(p-1)/2;k++)
       {
        if(Beta[x[2*k-1][i],x[2*k][i]==0)
```

```
        {
         min=0;
         break;
        }
       }
      S[i]=min;
      S[i]=Beta[S[i]][x[p][i]];
     }
   }
 }
```

El algoritmo contempla dos **for** anidados, uno que considera la dimensión n de los vectores y otro que considera todos los vectores del conjunto fundamental llamado p;

(PASO 2) (Vectores restringidos)

```
void paso2(void)
 {
  int u,i,j;
  nu=0;
  for(i=0;i<n;i++)
    if(S[i]==0)
      nu++;
  xu=new bool*[p];
  for(i=0;i<p;i++)
    xu[i]=new bool[nu];
  for(u=0;u<p;u++)
   {
    j=0;
    for(i=0;i<n;i++)
     {

      if(S[i]==0)
       {
        xu[u][j++]=x[u][i];
       }
     }
   }
 }
```

El algoritmo contempla dos **for** anidados, uno que considera la dimensión n de los vectores y otro que considera todos los vectores del conjunto fundamental llamado p;

(PASO 3) (Vectores negados)

```
void paso3(void)
 {
  int u,i,j;
```

```
  nx=new bool*[p];
  for(i=0;i<p;i++)
    nxu[i]=new bool[n];
  for(u=0;u<p;u++)
   {
    j=0;
    for(i=0;i<n;i++)
     {
      nx[u][i]=~x[u][i];
     }
   }
 }
```

El algoritmo contempla dos **for** anidados, uno que considera la dimensión n de los vectores y otro que considera todos los vectores del conjunto fundamental llamado p;

(PASO 4) (Vector soporte negado)

```
void paso4(void)
 {
  int i,k,min;
  for(i=0;i<n;i++)
   {
    min=1;
    if(p%2==0)
     {
      for(k=0;k<p/2;k++)
       {
        if(Beta[nx[2*k-1][i],nx[2*k][i]==0)
         {
          min=0;
          break;
         }
       }
      nS[i]=min;
     }
    else
     {
      for(k=0;k<(p-1)/2;k++)
       {
        if(Beta[nx[2*k-1][i],nx[2*k][i]==0)
         {
          min=0;
          break;
         }
       }
      nS[i]=min;
```

```
      nS[i]=Beta[nS[i]][nx[p][i]];
     }
   }
 }
```

El algoritmo contempla dos **for** anidados, uno que considera la dimensión n de los vectores y otro que considera todos los vectores del conjunto fundamental llamado p;

(PASO 5) (Vectores restringidos negados)

```
void paso5(void)
 {
  int u,i,j;
  nnu=0;
  for(i=0;i<n;i++)
    if(nS[i]==0)
      nnu++;
  nxu=new bool*[p];
  for(i=0;i<p;i++)
    nxu[i]=new bool[nnu];
  for(u=0;u<p;u++)
   {
    j=0;
    for(i=0;i<n;i++)
     {
      if(nS[i]==0)
       {
        nxu[u][j++]=nx[u][i];
       }
     }
   }
 }
```

El algoritmo contempla dos **for** anidados, uno que considera la dimensión n de los vectores y otro que considera todos los vectores del conjunto fundamental llamado p;

Definición 3.70 *Sean $f(n)$ y $g(n)$ dos funciones en $\mathbb{R}$. Se dice que $f(n)$ es $O(g(n))$ si existen dos enteros positivos a y b tales que $f(n) \leq a\ g(n)$ para cada $n \geq b$*

El siguiente teorema está tomado directamente del texto [107].

Teorema 3.71 *Sean $f_1(n)$ y $f_2(n)$ dos funciones tal que $f_1(n)$ es $O(g_1(n))$ y $f_2(n)$ es $O(g_2(n))$ se tiene:*

a) $f_1(n)+ f_2(n)$ es $O(max(g_1(n), (g_2(n))$

b) $f_1(n)f_2(n)$ es $O(g_1(n)(g_2(n))$

Analicemos la sentencia `for(int i=0; i<n; i++)`. La siguiente tabla muestra la cantidad de veces que se entra en el ciclo for dado un valor n.

n	`for(int i = 0; i < n; i + +)`
1	1
2	2
3	3
⋮	⋮
n	n

Tomemos a $f(n)$= `for(int i=0; i<n; i++)`. Afirmamos que $f(n)$ es $O(g(n))$. En efecto basta tomar $a = b = 1$ en la definición para O. Se tiene que

$$n = f(n) \leq a\ g(n) = 1\, g(n) = 1\, n \text{ para cada } 1 = b \geq n$$

Por lo tanto $f(n)$ es $O(g(n)$, es decir, un **for** simple tiene orden de complejidad lineal.

Ahora, consideremos dos **for** anidados, puesto que el orden de cada **for** es lineal, por el inciso b del teorema anterior se concluye que el **for** anidado es cuadrático. A continuación se mostrará la complejidad por fases de cómo quedó finalmente después de haber realizado el análisis.

Fase de Aprendizaje

Paso	`Complejidad`
1	$O(n^2)$
2	$O(n^2)$
3	$O(n^2)$
4	$O(n^2)$
5	$O(n^2)$

Fase de Recuperación

Paso	`Complejidad`
1	$O(n)$
2	$O(n^2)$
3	$O(n^2)$
4	$O(n^2)$
5	$O(n^2)$
6	$O(n)$
7	$O(n)$
8	$O(n^2)$
9	$O(n^2)$
10	$O(n^2)$
11	$O(n^2)$
12	$O(n)$
13	$O(n)$

De la primera tabla se puede observar que se tiene $O(n^2 + n^2 + n^2 + n^2 + n^2) = 5\ O(n^2)$, esto representa que efectivamente la fase de aprendizaje tiene un costo de $O(n^2)$.

En la segunda tabla arroja $O(n+n^2+n^2+n^2+n^2+n+n+n^2+n^2+n^2+n^2+n+n) = 5(O(n)) + 8(O(n^2))$, lo que también esto nos sugiere que la fase de recuperación tiene un costo de $O(n^2)$.

Cabe aclarar que las operaciones se realizaron bit a bit entre vectores.

Finalmente se concluye que tanto la fase de aprendizaje como la fase de recuperación son de $O(n^2)$.

Capítulo 4

Resultados y Discusión

Este Capítulo está dedicado a la presentación de resultados experimentales realizados con el algoritmo propuesto y su discusión. Los experimentos se realizaron con un *software* diseñado e implementado en Borland C++ Builder 6.0, *ex-profeso* para este fin. Dicho *software* se ejecutó en una computadora personal (armada) con microprocesador DualCore Intel Pentium D 950 de 64 bits a 3400 MHz, 2 GBytes de memoria DDR2 SDRAM, con disco duro SCSI de 80 GBytes; sobre el sistema operativo Windows XP Professional x64 bits.

Dos bases de datos se escogieron para probar experimentalmente el nuevo algoritmo de máquinas asociativas Alfa-Beta con soporte vectorial: la base de datos MNIST de caracteres escritos a mano, motivo de la sección 4.1, y la base de datos *Iris Plants*, cuya descripción y resultados experimentales se presentan en la sección 4.2.

4.1. Base de datos MNIST de dígitos escritos a mano

La base de datos MNIST de dígitos escritos a mano es un subconjunto de imágenes tomadas de una base de datos más extensa que proporciona el NIST (National Institute of Standards and Technology): fue construida a partir de la Base de Datos Especial 3 y Base de Datos Especial 1 del NIST (Instituto Nacional de Estándares y Tecnología de los Estados Unidos, por sus siglas en inglés), que contienen imágenes de dígitos escritos a mano [93].

Dos son las razones principales por las cuales se decidió tomar esta base de datos para la fase experimental del presente trabajo de tesis:

- Por sus características, los patrones de esta base de datos se prestan de manera muy clara, como se explicó ampliamente en la sección 3.1, para probar el algoritmo de las máquinas asociativas Alfa-Beta con soporte vectorial.
- Diversos grupos de investigación reconocidos en el mundo la han utilizado para evaluar el desempeño de sus algoritmos de aprendizaje, permitiendo hacer una comparación consistente entre los resultados presentados en esta tesis y los que se encuentran actualmente en la literatura científica relacionada.

La descripción de la la base de datos MNIST y los resultados experimentales relacionados con el modelo de máquinas asociativas Alfa-Beta con soporte vectorial, se presentan en cinco partes.

Primeramente se describe la base de datos y la manera en que los patrones están etiquetados en 10 clases diferentes. Acto seguido, se describen brevemente dos métodos de prueba del algoritmo con los patrones sin preprocesamiento; es decir, usando los valores originales de los pixelees en escala de grises.

Las dos partes siguientes constan de dos sendos métodos de umbralización, que permiten trabajar con imágenes binarias; mientras que la tercera parte consiste en la explicación de un método de umbralización a través de promedios calculados en una ventana de 3×3 pixeles, en la cuarta parte se describe un método de umbralización a través del uso de histogramas, a la manera de Otsu [104].

Finalmente, en la quinta parte se presenta un resumen de los resultados experimentales, y la correspondiente discusión.

4.1.1. Las diez clases de la base de datos MNIST

La base de datos MNIST consiste en un conjunto de 70,000 imágenes agrupadas en 10 clases diferentes, de las cuales 60,000 instancias son utilizadas para el entrenamiento del algoritmo y las 10,000 restantes para prueba. Cada una de las instancias de esta base de datos está normalizada y centrada dentro de una imagen en tonos de gris de 28×28 pixeles; las figuras 4.1 a 4.4 son ejemplos de patrones de entrenamiento (dígitos escritos a mano) para las clases 0,1,2 y 9 respectivamente.

Nota 4.1 *No obstante que en la base de datos original los tonos más oscuros se acercan al cero y los tonos más claros están cerca del valor 255, para efectos de mejor visualización, en todos los experimentos de esta tesis se han tomado los negados de los patrones, de modo que el fondo sea blanco y el carácter sea más cercano al negro.*

Figura 4.1: Patrones de entrenamiento para la clase 0

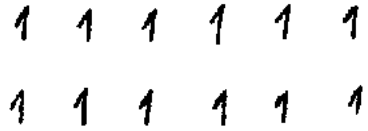


Figura 4.2: Patrones de entrenamiento para la clase 1

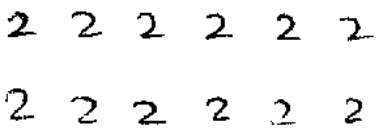


Figura 4.3: Patrones de entrenamiento para la clase 2

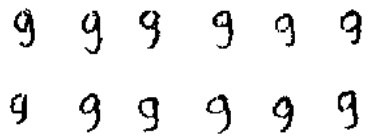


Figura 4.4: Patrones de entrenamiento para la clase 9

Aun cuando la tarea que se está llevando a cabo durante la fase de prueba es la de recuperar o reconocer patrones, por la naturaleza de la base de datos empleada, es posible atacar el problema de reconocimiento de dígitos escritos a mano como un problema de clasificación multiclase. Para ilustrar lo anterior veamos un ejemplo.

Ejemplo 4.2 *Dado el conjunto de instancias disponibles para entrenamiento en la base de datos MNIST; tomar cada una de las 60,000 imágenes y formar las asociaciones correspondientes, posteriormente, tomar las 10,000 instancias de prueba y proceder con su recuperación..*

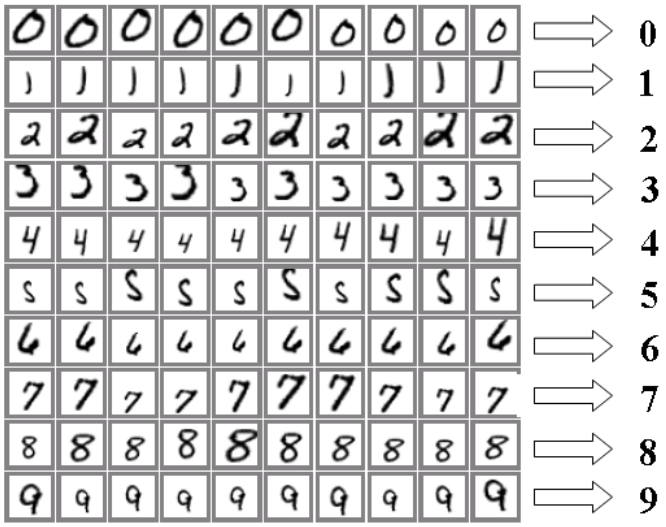


Figura 4.5: Conjunto de entrenamiento

Si tomamos cada una de las 60,000 instancias de entrenamiento y a cada una le asociamos la etiqueta de clase que le corresponde, obtendremos asociaciones como las que se muestran de la figura 4.5.

Del mismo modo, al ejecutar el algoritmo sobre las 10,000 instancias de prueba se espera obtener sin ambiguedad la etiqueta de clase que le corresponde a cada patrón; consecuentemente, para todos aquellos patrones de prueba que entreguen la misma etiqueta de clase, se les puede considerar elementos de una misma clase. Así, es posible atacar el problema de reconocimiento de dígitos escritos a mano como un problema de clasificación multiclase. La figura 4.6 muestra una ventana de 10 instancias de entrenamiento con sus respectivas etiquetas de clase, mientras que la figura 4.7 muestra una ventana de 60 instancias de prueba con sus respectivas etiquetas de clase recuperadas correctamente.

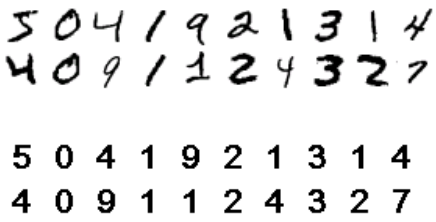


Figura 4.6: Ventana de 10 instancias de entrenamiento

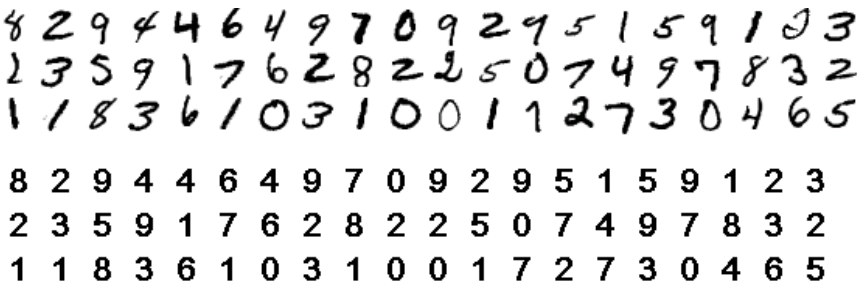


Figura 4.7: Ventana de 60 instancias de prueba

4.1.2. Dos experimentos sin preprocesamiento

En virtud de que el modelo de las máquinas asociativas Alfa-Beta con soporte vectorial requiere de patrones binarios para su funcionamiento en ambas fases, y dado que los patrones de la base de datos MNIST son imágenes en escala de grises, es necesario realizar una codificación de los valores de cada pixel, para obtener las cadenas binarias correspondientes.

En el primer experimento la codificación se realizó a la manera usual: convirtiendo directamente cada valor de pixel, el cual está entre 0 y 255, en una cadena binaria normal usando el conocido algoritmo de conversión binaria [103].

Para el segundo experimento, se utilizó un método de conversión más sofisticado: el código Johnson-Möbius modificado [56].

Los resultados de ambos experimentos se presentan en la tabla 4.1.

4.1.3. Método de umbralización basado en promedios locales

A continuación se presenta un método para obtener automáticamente un valor de umbral para el proceso de binarización de las imágenes en escala de gris que conforman la base de datos MNIST. La idea que subyace a este método es tomar información local de los vecinos de un pixel para elegir el umbral para ese pixel.

Primero, se forma una ventana de 3×3 pixeles; con esta ventana se tomará la información de los 8 vecinos contiguos al pixel central, y se obtiene el promedio de los valores de los pixeles vecinos; este valor constituye el valor del umbral para el pixel central. Ahora se compara el valor del pixel que se está trabajando, el pixel central de la ventana, con el valor de umbral obtenido anteriormente: si el valor actual es mayor o igual al umbral, se asigna el valor correspondiente al *fondo* (que normalmente es el valor lógico 0); en caso contrario, se asigna el valor del *objeto* (que normalmente es el valor lógico 1). Este nuevo valor binario se asigna al pixel correspondiente de la imagen binarizada. Finalmente se recorre la ventana por todos y cada uno de los pixeles de la imagen original.

Más específicamente, sea una imagen I de N pixeles en escala de grises cuyos valores de pixel son $\{1, 2, 3, ..., L\}$, con el valor fijo $L \in \mathbb{Z}^+$. Representemos el valor de un pixel cualquiera de la imagen, el cual tiene coordenadas (x, y) por p_0, y consideremos que las coordenadas y los valores de sus $8-$vecinos están distribuidos en la vecindad de Moore de la siguiente manera:

Coordenadas	Valor de pixel
$(x+1, y)$	p_1
$(x+1, y-1)$	p_2
$(x, y-1)$	p_3
$(x-1, y-1)$	p_4
$(x-1, y)$	p_5
$(x-1, y+1)$	p_6
$(x, y+1)$	p_7
$(x+1, y+1)$	p_8

La distribución espacial de los valores del pixel (x, y) y de sus $8-$vecinos luce así:

p_6	p_7	p_8
p_5	p_0	p_1
p_4	p_3	p_2

(4.1)

El umbral u que se utiliza en este método es el promedio local de los $8-$vecinos

$$u = \frac{1}{8}\sum_{i=1}^{8} p_i \tag{4.2}$$

Este método de umbralización asigna un valor a p_0 de acuerdo con la siguiente expresión:

$$p_0 = \begin{cases} 1, \text{ si } p_0 \geq u \\ \\ 0, \text{ si } p_0 < u \end{cases} \tag{4.3}$$

Nota 4.3 *En los bordes y las esquinas, a los vecinos que quedan fuera de la imagen se les asigna el valor* 0.

Con objeto de aplicar el método de umbralización basado en promedios locales a la imagen I, se localiza el primer pixel I y ahí se centra la vecindad de Moore 4.1 tomando en consideración la nota 4.3. Acto seguido, se calcula el umbral u mediante la fórmula 4.2 y se aplica la umbralización 4.3. Se hace lo mismo para todos los pixeles de la imagen I.

Para mayor claridad, veamos un ejemplo.

Ejemplo 4.4 *Sea la imagen de la figura 4.8 la imagen original a ser binarizada. Seguir el método descrito arriba para obtener la imagen binarizada.*

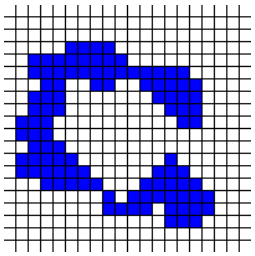


Figura 4.8: Imagen a ser binarizada. Un pixel blanco representa un valor alto, mientras que un pixel azul representa un valor bajo, ambos en la escala de grises

Figura 4.9: Ventana de 3×3

Nótese que los pixeles blancos de la imagen representan valores altos (cercanos al 255, el color blanco en la escala de grises) y los pixeles azules representan valores bajos

(cercanos al 0, el color negro en la escala de grises). Entonces, el primer paso es formar la ventana de 3×3 pixeles; ésta se puede ver en la figura 4.9.

Ahora, dado un pixel, se centra la ventana en dicho pixel y se procede a calcular el promedio de los valores de los pixeles vecinos, marcados por la ventana, y asignarlo como el umbral correspondiente.

Un ejemplo de lo anterior se puede apreciar en la figura 4.10, en donde el pixel en cuestión está marcado en verde. Como indica el método, lo que sigue es comparar el valor del umbral con valor del pixel. En este caso, se tienen 4 vecinos con valores altos (pixeles blancos) y 4 vecinos con valores bajos (pixeles negros), lo que da un promedio cercano al valor medio de la escala, 128. Es claro que este valor será mayor que el valor de los pixeles bajos, por lo que se le asigna un valor de *objeto* (valor lógico 1) al pixel.

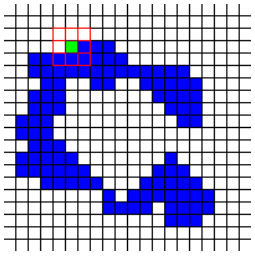


Figura 4.10: Ventana centrada en el pixel $(6, 4)$ de la imagen

Lo que sigue es recorrer la imagen para trabajar con cada pixel. En este sentido, existen varios casos particulares de interés. Primero, ¿qué sucede con un pixel del fondo, rodeado de pixeles de fondo? Esto se puede ver en la figura 4.11. Dado que en este caso todos los pixeles presentes en la ventana tienen un valor muy similar, cercano al blanco, el promedio de los pixeles de fondo dará valores altos. Asumiendo que todos estos pixeles tengan un valor igual o muy cercano (algo bastante común en este tipo de imágenes), es de esperarse que el valor del pixel central sea igual o mayor que el promedio, así que al pixel se le asigna un valor de *fondo* (valor lógico 0).

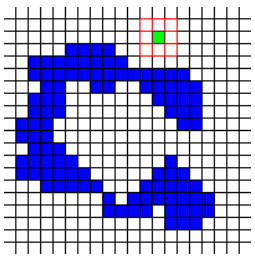


Figura 4.11: Caso de interés: pixel del fondo rodeado de pixeles de fondo

Veamos ahora el caso contrario: ¿qué sucede con un pixel del objeto, rodeado de pixeles de objeto? Esto se puede ver en la figura 4.12. La situación es muy similar al caso

anterior; todos los pixeles presentes en la ventana tienen un valor muy similar, cercano al negro, por lo que el promedio dará valores bajos. Tomando la misma asunción del caso anterior, que todos estos pixeles tengan un valor igual o muy cercano, se espera que el valor del pixel central sea igual o mayor que el promedio, así que al pixel se le asigna un valor de *fondo* (valor lógico 0). Esto parece un tanto desconcertante... sin embargo, el efecto que tiene en el proceso de umbralización es que, en la mayoría de las imágenes de la base de datos MNIST, este método da valores binarios de *objeto* (valor lógico 1) al *contorno* del mismo, y valor de *fondo* (valor lógico 0) al resto de la imagen.

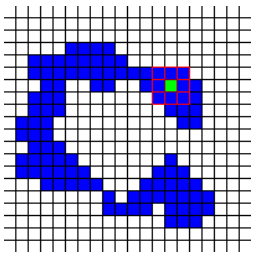


Figura 4.12: Caso de interés: pixel del objeto rodeado de pixeles de objeto

El tercer caso de interés se presenta en la frontera de la imagen misma: para poder centrar en la ventana los pixeles de la orilla, algunos de los pixeles vecinos circunscritos por la ventana quedan "fuera de la imagen"(ver figura 4.13). Existen básicamente dos enfoques de solución para esta situación. El primero es asignar un valor predeterminado, elegido arbitrariamente, a estos pixeles "fantasma". La otra opción es ignorarlos para el cálculo de los promedios, tomando en cuenta sólo aquellos que quedan dentro de la imagen. Si se elige la primera opción, lo ideal es asignar un valor cercano a los valores de los pixeles que se encuentran en la frontera; en el caso de la figura 4.13, convendría un valor alto. Si se elige la segunda opción, en la figura 4.13 participarían sólo lo pixeles $(1, 2)$, $(2, 1)$ y $(2, 2)$ para calcular el umbral del pixel $(1, 1)$. En el presente trabajo se utilizó la primera opción, asignándole el valor 255 (blanco) a los pixeles fuera de la imagen.

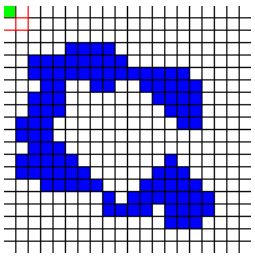


Figura 4.13: Caso de interés: pixel de la orilla de la imagen

Entonces, el resultado final de la binarización de la imagen 4.8 se muestra en la figura 4.14. En este caso se representan los pixeles de *fondo* (valor lógico 0) con el color blanco y los pixeles de *objeto* (valor lógico 1) con el color verde.

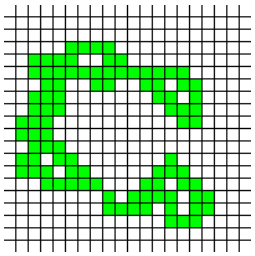


Figura 4.14: Resultado de la binarización

Para llevar a cabo el experimento, se realizó una etapa de preprocesamiento a todas y cada una de las 70,000 imágenes de la base de datos, y para ello se aplicó el método de umbralización aquí descrito. Acto seguido, se trabajaron ambas fases del modelo de las máquinas asociativas Alfa-Beta con soporte vectorial con las imágenes binarias obtenidas.

Los resultados del experimento se presentan en la tabla 4.1.

4.1.4. Método de umbralización de Otsu [104]

En 1979, Otsu propuso un método de umbralización de imágenes en escala de grises bimodales, basado en una idea simple: buscar un umbral que minimice la varianza dentro de cada una de las clases de pixeles, al mismo que maximiza la varianza entre las dos clases.

CONSIDERACIONES:

- Sea una imagen I en escala de grises que deberá ser umbralizada, y quedará con dos clases de pixeles: la clase, la 1 y la clase 2
- N es el número total de pixeles de I
- n_i es el número de pixeles con valor $i = 0, 1, 2, 3, ..., 255$
- La probabilidad del valor i se calcula así:

$$p(i) = \frac{n_i}{N}$$

- Si t es un umbral tal que $t = 0, 1, 2, 3, ..., 255$, $q_1(t)$ y $q_2(t)$ se calculan así:

$$q_1(t) = \sum_{i=0}^{t} p(i)$$

$$q_2(t) = \sum_{i=t+1}^{255} p(i)$$

ALGORITMO DE OTSU:
Inicialización
1)

$$q_1(0) = p(0)$$

2)

$$\mu_1(0) = 0$$

3)

$$\mu = \sum_{i=0}^{255} i \cdot p(i)$$

Recursión
1)

$$q_1(t+1) = q_1(t) + p(t+1)$$

2)

$$\mu_1(t+1) = \frac{q_1(t)\mu_1(t) + (t+1)p(t+1)}{q_1(t+1)}$$

3)

$$\mu_2(t+1) = \frac{\mu - q_1(t+1)\mu_1(t+1)}{1 - q_1(t+1)}$$

Final
Buscar el valor de t que maximice la expresión:

$$\sigma_B^2(t) = q_1(t)\left[1 - q_1(t)\right]\left[\mu_1(t) - \mu_2(t)\right]^2$$

Salida: t es el umbral buscado.

4.1.5. Método de umbralización basado en histogramas

Inicialmente este experimento se llevó a cabo realizando binarizaciones con umbrales propuestos manualmente por el usuario del software. En esos intentos de ensayo y error, se encontró que un buen valor de umbral estaba entre 10 y 12.

Sin embargo, se inició una evolución del experimento con el intento de automatizar la determinación del umbral. La hipótesis es similar a la de Otsu: el método es útil para umbralizar imágenes bimodales en escala de grises (comúnmente de 256 valores).

La idea es muy simple, y se resume en los siguientes pasos:

- Se suman todas las ocurrencias de un valor dado de pixel, para todos los patrones de la base de datos.
- Se traza un histograma acumulado de todas las sumas.
- Se localiza el par de valores (entre 0 y 255) para el cual el cambio es cero; o el trío de valores para los cuales la tendencia cambia.

Más concretamente, sea una imagen I en escala de grises que deberá ser umbralizada, y quedará con dos clases de pixeles: la clase, la 1 y la clase 2. Sea N es el número total de pixeles de I, y representemos por n_i el número de pixeles con valor $i = 0, 1, 2, 3, ..., 255$.

Después de trazar el histograma acumulado,consideremos los puntos en el plano: $(0, n_0)$, $(1, n_1)$, $(2, n_2)$, $(3, n_3)$, $(4, n_4)$, ..., $(255, n_{255})$, en ese orden.

Representemos por m_i la pendiente del segmento de recta que une los dos puntos (i, n_i) y $(i + 1, n_{i+1})$; esa pendiente se calcula así:

$$m_i = \frac{n_{i+1} - n_i}{i + 1 - i} = \frac{n_{i+1} - n_i}{1} = n_{i+1} - n_i \tag{4.4}$$

ALGORITMO

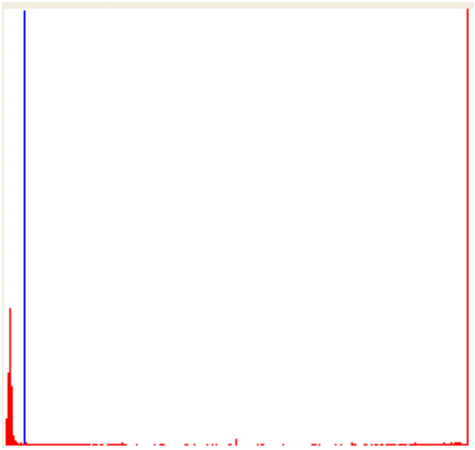
- Iniciar con $i = 0$
- Poner un contador que se inicia en $c = 0$
- Repetir lo siguiente:

{
Calcular m_i
Calcular m_{i+1}
Si {[((signo de m_i es +)OR($m_i = 0$))AND(signo de m_{i+1} es −)]OR[((signo de m_i es −)OR($m_i = 0$))AND(signo de m_{i+1} es +)]}incrementar el contador c
}
Hasta que [($m_i = 0$ AND $c = 1$) OR $c = 2$]
Final: el umbral que se busca es i

Al aplicar el algortimo la base de datos MNIST, el histograma acumulado se construyó con los 60,000 patrones de aprendizaje. El primer incremento al contador c se dio en el par de valores $i = 1$, $i = 2$; y el segundo incremento se dio en el par de valores $i = 6$, $i = 7$, por lo que aquí se da la salida del algoritmo con el valor de umbral $i = 6$.

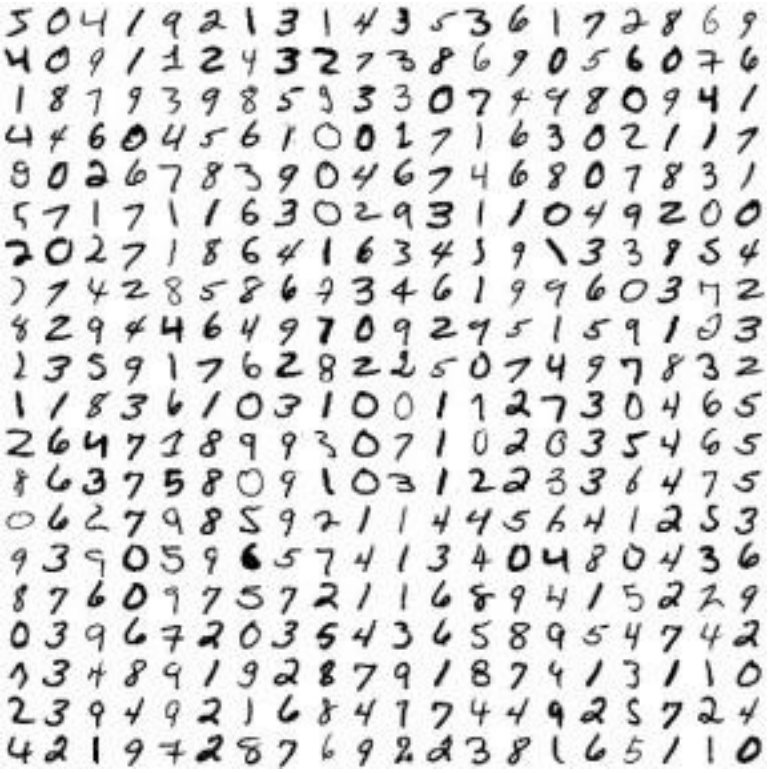
Los primeros 13 valores se muestran a continuación, y más abajo aparece el histograma de esos totales.

0	314,282
1	801,557
2	1,513,207
3	653,888
4	117,808
5	59,559
6	39,342
7	35,418
8	37,941
9	33,844
10	35,397
11	37,199
12	37,029

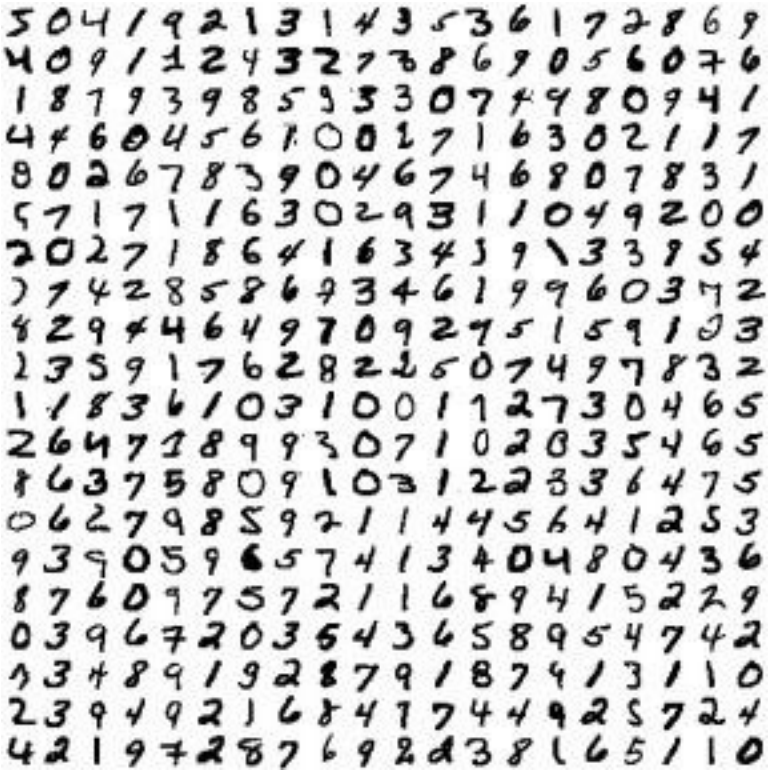


Los resultados del experimento para un cálculo automático del umbral se presentan en la tabla 4.1.

Un ejemplo de las imágenes con las que se trabajó, tanto las originales en escala de grises como las binarizadas, se puede apreciar en la figura 4.15.



a) Imágenes en escala de grises



b) Imágenes binarias

Figura 4.15. Base de datos MNIST

4.1.6. Cuadro de resultados

Los resultados obtenidos se comparan con los resultados presentados en [94] a [99], mismos que se muestran en la tabla 4.1 (incluyendo, por supuesto, al algoritmo propuesto en este trabajo de tesi en todas sus modalidades)

Nota 4.5 *En el Cuadro 4.1 usaremos la abreviatura SVM Alfa-Beta para referirnos a las máquinas asociativas Alfa-Beta con soporte vectorial.*

Cuadro 4.1: Comparación entre varios algortimos aplicados a la base de datos MNIST

Algoritmo [Referencia]	Año	Preprocesamiento	Error
k-NN, Euclidean (L2) [94]	1998	ninguno	5.0 %
k-NN, Euclidean (L2) [94]	1998	deskewing	2.4 %
SVM Alfa-Beta (codificación binaria)	**2008**	**ninguno**	**2.1 %**
2-layer NN, 300 HU [94]	1998	deskewing	1.6 %
2-layer NN, 800 HU, Cross-Entropy [95]	2003	ninguno	1.6 %
2 layer MLP (CE) [95]	2003	ninguno	1.6 %
SVM [96]	2002	affine	1.4 %
k-NN, Tangent Distance [94]	1998	affine & thick	1.1 %
2 layer MLP (CE) [95]	2003	affine	1.1 %
k-NN, Tangent Distance [94]	1998	subsampling 16x16 16x16 pixeles	1.1 %
NN, RBM + NCA training [97]	2007	ninguno	1.0 %
Convolutional net LeNet-5 [94]	1998	ninguno	0.95 %
2-layer NN, 800 HU, MSE [95]	2003	distortiones elásticas	0.9 %
large conv. net, random features [98]	2007	ninguno	0.89 %
Convolutional net LeNet-5 [94]	1998	distortions enormes	0.85 %
SVM Alfa-Beta (umbralización - promedios locales)	**2008**	**binarizado**	**0.66 %**
SVM Alfa-Beta (umbralización - Otsu)	**2008**	**binarizado**	**0.65 %**
SVM Alfa-Beta (código Johnson-Möbius modificado)	**2008**	**ninguno**	**0.52 %**
SVM Alfa-Beta (umbralización - histogramas)	**2008**	**binarizado**	**0.49 %**
large conv. net, unsup pretraining [99]	2006	distortiones elásticas	0.39 %

Se puede observar que las máquinas asociativas Alfa-Beta con soporte vectorial tienen un desempeño competitivo con respecto a otros algoritmos de la literatura científica actual.

4.2. Base de datos *Iris Plants*

A continuación se presenta la base de datos *Iris Plants Database*, que fue tomada del Repositorio de *Machine Learning* del Departamento de Información y Ciencias de

la Computación de la University of California, Irvine [105]. Esta es una base de datos clásica en el área de Reconocimiento de Patrones y, a pesar de ser una de las primeras bases de datos utilizadas para probar algoritmo del área, sigue siendo muy utilizada. El conjunto de datos contiene 3 clases de 50 instancias cada una, donde cada clase hace referencia a un tipo de planta de iris (Iris Setosa, Iris Versicolor e Iris Virginica). Cada patrón que representa una planta de iris tiene 4 atributos más uno que representa la clase. En esta base de datos los rasgos que representan el largo y ancho de los pétalos están altamente correlacionados. Asimismo, se conoce que mientras la clase 1 (Iris Setosa) es linealmente separable de las otras dos clases, las clases 2 y 3 (Iris Versicolor e Iris Virginica, respectivamente) no son linealmente separables entre sí. En el Cuadro 4.2 se describen los rasgos presentes en esta base de datos.

Cuadro 4.2. Rasgos de la base de datos *Iris Plant*

Rasgo	**Descripción**
1	Largo del sépalo en centímetros
2	Ancho del sépalo en centímetros
3	Largo del pétalo en centímetros
4	Ancho del pétalo en centímetros
5	Clase

El Cuadro 4.3 se presenta una comparación de los redimientos reportados en la tesis [52] y en un artículo publicado el mes de enero de 2007 en la prestigiosa revista Pattern Recognition [106], del desempeño de varios clasificadores con respecto a la base de datos *Iris Plant*. En particular, es importante hacer notar que los dos mejores algoritmos según la Tabla 6.4 de la mencionada tesis, al tomar los mejores 20 resultados de 1000 experimentos, son el 1-NN con la mejor clasificación correcta de 96.67 % y el CHAT con la mejor clasificación correcta de 98.67 %. Por otro lado, los dos mejores clasificadores multiclase presentados en la Tabla 7 del artículo mencionado, son el de 3 redes neuronales construido con el método OAO y el de 1 red neuronal construido con el método OAA, ambos con la mejor clasificación correcta de 98 % al probarlos con un proceso de *10-fold cross validation*.

Cuadro 4.3. Comparación del rendimiento con la base de datos *Iris Plant*

Clasificador	**Mejor rendimiento**
CHAT	98.67 %
OAO 3 nets	98 %
OAA 1 net	98 %
1-NN	96.67 %
C-means difuso usando func. de disimilaridad	96.0 %
OAA 3 nets	96 %
OAHO 2 nets	96 %
3-NN	95.33 %
C-means	89.33 %
C-means difuso	90.0 %
SVM Alfa-Beta (codificación binaria)	**99.3 %**

Es notable que, no obstante que el tipo de datos de esta base de datos no es el más adecuado para ser trabajado por las máquinas asociativas Alfa-Beta con soporte vectorial, el resultado fue el mejor de los comparados.

Capítulo 5

Conclusiones y Trabajo Futuro

En este capítulo se presentan las conclusiones derivadas de los resultados obtenidos en el proceso de este trabajo de tesis; además, se proponen algunos de los posibles trabajos que se podrían realizar con objeto de continuar con las ideas propuestas aquí, tanto en lo que resta del proceso propio de este trabajo de tesis, como en general, dando la pauta a futuros investigadores sobre los puntos no cubiertos, pero que pudieran ser afrontados en otros trabajos de investigación.

5.1. Conclusiones

1. En este trabajo de tesis se introduce un nuevo modelo matemático de reconocimiento automático de patrones: las máquinas asociativas Alfa-Beta con soporte vectorial.

2. Se define un vector de soporte para el nuevo modelo, el cual es diferente del concepto de vector de soporte de las SVM clásicas. El nuevo vector de soporte se forma con la información redundante al considerar la totalidad de los patrones del conjunto fundamental.

3. Se definen dos transformadas: la transformada τ de x con respecto a y que permite calcular la alteración aditiva de x con respecto a y; y la transformada τ de x con respecto a y que permite calcular la alteración total del vector x con respecto al vector y.

4. El modelo de reconocimiento de patrones propuesto es competitivo, en rendimiento, con respecto a los algoritmos presentes en la literatura científica actual; en particular al clasificar caracteres escritos a mano tomados de la base de datos MNIST.

Referencias

[1] Sánchez-Garfias, F.A., Díaz-de-León, J.L., Yáñez-Márquez, C.: Reconocimiento automático de patrones. Conceptos básicos, IT 79, Serie Verde, Centro de Investigación en Computación, IPN, México (2003)

[2] Marqués de Sá, J.P.: Pattern Recognition, Concepts, Methods and Application. Germany, Springer (2001)

[3] Duda, R.O., Hart, P.E., Stork, D.G.: Pattern Classification. John Wiley & Sons, USA.(2001)

[4] Aldape-Pérez, M., Yáñez-Márquez, C. , López -Leyva, L.O.: Feature Selection using a Hybrid Associative Classifier with Masking Technique. en: IEEE Computer Society, Proc. Fifth Mexican International Conference on Artificial Intelligence, MICAI 2006 pp. 151-160. ISBN: 0-7695-2722

[5] Webb, A.: Statistical Pattern Recognition. Oxford University Press, USA (1999)

[6] Kuncheva, L.I.: A theoretical Study on Six Classifier Fusion Strategies. IEEE Transations on Pattern Analysis and Machine Intellegence, vol. 24, no. 2 (2002) 281-286

[7] Santiago-Montero, R., Yáñez-Márquez, C., Diaz-de-León, J. L.: Clasificador híbrido de patrones basado en la Lenmarix de Steinbuch y el Linear Associator de Anderson-Kohonen. Research on Computing Science. Reconocimiento de patrones, avances y perspectivas. Centro de Investigación en Computación, IPN, México (2002) 449-460

[8] Sánchez-Garfias, F.A: Condiciones necesarias y suficientes para recuperación perfecta de patrones. Lernmatrix de Steinbuch. Tesis de Maestría en Ciencias de la Computación. CIC IPN, México (2004)

[9] Friedman, M., Kandel, A.: Introduction to Pattern Recognition (Statistical, Structural, Neural and Fuzzy Logic Approaches). Singapore, World Scientific (2000)

[10] Sarünas, R.: Statistical and Neural Classifiers, An integrated Approach to disign. MIT Press, England (2001)

[11] Schürmann, J.: Pattern classification, A unified view of statistical and neural approaches. John Wiley, USA.(1996)

[12] Shalkoff, R.: Pattern recognition, Statistical, Structural and Neural Approaches. John Wiley, USA.(1992)

[13] Michie,D., D.J. , Spiegelhalter, Taylor, C.C.: Machine Learning, Neural and Statistical Classification. Ellis Horwood, Chichester, England (1994)

[14] Fisher, R.A.: The use of multiple measurements in taxonomic problems. Annual Eugenics,vol. 7, part II (1936) 179-188

[15] Lu, Y., Tan, C.L.: Combination of multiple classifiers using probabilistic dictionary and its application to postcode recognition, Pattern Recognition, vol. 35 (2002) 2823-2832

[16] Rueda, L., Oommen, B.J.: On optimal pairwise linear classifiers for normal distributions the two-dimensional case. IEEE Transations on Pattern Analysis and Machine Intellegence, vol. 24, no. 2 (2002) 274-273

[17] Díaz-de-León, J.L., Yáñez-Márquez, C., Sánchez-Garfias, F.A.: Reconocimiento de patrones. Enfoque probabilístico-estadístico. IT 83, Serie Verde, Centro de Investigación en Computación, IPN, México (2003)

[18] Cover, T.M., Hart, P.E.: Nearest Pattern Classification. IEEE Trans. on Information Theory, vol. IT-13, no. 1 (1967) 21-27

[19] Bandyopadhyay, S., Maulik, U.: Efficient prototype reordering in nearest neighbor classifation. Pattern Recognition, vol. 35 (2002) 2791-2799

[20] Dasarathy, B.V.: Nearest Neighbor (NN) Norms: NNpattern Classification Techniques. IEEE Computer Society Press, USA (1991)

[21] Demeniconi, C., Peng, J., Gunopulos, D.: Locally Adaptive Metric Nearest-Neighbor Classification. IEEE Transations on Pattern Analysis and Machine Intellegence, vol. 24, no. 9 (2002) 1281-1285

[22] Ho, S.Y., Liu, C.C., Liu, S.: Design of an optimal nearest neighbor classifier using an intelligent genetic algorithm. Pattern Recognition Letters, vol. 23 (2002) 1495-1503

[23] Huang, Y.S., Chiang, C.C., Shieh, J. W., Grimson, E.: Prototype optimization for nearest-neighbor classification. Pattern Recognition, 35 (2002), 1237-1245

[24] Wu, Y., Ianekiev, K., Govindaraju, V.: Improved k-nearest neighbor classification. Pattern Recognition, vol. 35 (2002) 2311-2318

[25] Díaz-de-León, J.L., Yáñez-Márquez, C., Sánchez-Garfias, F.A.: Clasificador euclideano de patrones. IT 80, Serie Verde, Centro de Investigación en Computación, IPN, México (2003)

[26] Flores Carapia, R., Yáñez Márquez, C.: Minkowski's Metrics-Based k-NN Classifier Algorithm: A Comparative Study. Research on Computing Science, Vol. 14, IPN México (2005) pp. 191-202. ISSN 1665-9899

[27] Flores Carapia, R., Yáñez Márquez, C.: Minkowski's Metrics-Based Classifier Algorithm: A Comparative Study. en: Memoria del XIV Congreso Internacional de Computación CIC IPN, México (2005) pp. 304-315. ISBN: 970-36-0267-3

[28] Muñoz Torija, J.M.,Yáñez Márquez, C: Un Estudio Comparativo del Perceptron y el Clasificador Euclideano, en: Memoria del XIV Congreso Internacional de Computación CIC IPN México (2005) pp. 316-326. ISBN: 970-36-0267-3

[29] Anil, K. Jain, Robert, P. W., Duin, Mao, Jianchang Mao: Statistical Pattern Recognition. A Review:. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, num. 1 (2000) 4-37

[30] Yáñez-Márquez, C., Diaz de León, J.L., Sánchez-Garfias, F.A.: Reconocimiento de patrones. Enfoque sintáctico-estructural. IT 84, Serie Verde, Centro de Investigación en Computación, IPN, México (2003)

[31] Gonzalez, R.C., Thomason, M.G.: Syntactic Pattern Recognition: an Introduction. Addison Wesley (1982)

[32] McCulloch, W., Pitts, W.: A logical calculus of the ideas immanent in nervous activity. Bulletin of Mathematical Biophysics, vol. 5 (1943) 115-133

[33] Rosenblatt, F.: The Perceptron: A probabilistic model for information storage and organization in the brain. Psychological Review, vol. 65 (1958) 386-408

[34] Pal, S.: Neuro-Fuzzy, Pattern Recognition: Methods in Soft Computing. USA, John Wiley & Sons (1999)

[35] Pandya, A.S.: Pattern recognition with neural networks in C++. Springer-Verlag, Great Britain (1996)

[36] Abe, S.: Pattern classification, Neuro-Fuzzy Methods and their Comparison. Springer-Verlag, Grat Britain (2001)

[37] Acharya, U.R., Bhat, P.S., Iyengar, S.S., Rao, R., Dua, S.: Classification of heart rate data using artificial neural network and fuzzy equivalence relation. Pattern Recognition, vol. 36, issue 1 (2003) 61-81

[38] Hopfield, J.J.: Neural networks and physical systems with emergent collective computational abilities. Proceedings of the National Academy of Sciences, vol. 79 (1982) 2554-2558

[39] Hopfield, J.J.: Neurons with graded respose have collective computational properties like those of two-state neurons. Proceedings of the National Academy of Sciences, vol. 81 (1984) 3088-3092

[40] Anderson, J.A., Rosenfeld, E(eds.): Neurocomputing: Fundations of Research. MIT Press, Cambridge (1990)

[41] Anderson, J.A., Silverstein, J., Ritz, S., Jones, R.: Distinctive features, categorical perception, and probability learning: some applications of a neural model. Psicological Review, vol. 84 (1977) 413-451

[42] Haykin, S.: Neural Networks, A Comprehensive Foundation. Prentice Hall, USA (1999)

[43] Hassoun, M.H.: Fundamentals of Artificial Neural Networks. MIT Press, Cambridge (1995)

[44] Kishan, M., Chilukuri, K.M., Sanjay, R.: Elements of Artificial Neural Networks. MIT Press, USA (1997)

[45] Minsky, M., Papert, S.: Perceptrons. MIT Press, Cambridge (1969)

[46] Rumelhart, D.E., Hinton, G. E., Williams, R.J.: Learning internal representation by Backpropagating errors. Nature, 323 (1986) 533–536

[47] Lecun, Y.: Une Procedure d´apprentissage pour reseau a seuil assymetrique cognitiva 85 : A la Frontiere de l'Intelligence Artificielle des Sciences de la Connaissance des Neurosciences, Paris, France. (1985) 559–604

[48] Krauth, W., Mezard, M.: Learning algorithms with optimal stability in neural networks. J. Phys. A: Math. Gen, vol. 20:l745 1987

[49] F. Rosenblatt.: Principles of Neurodynamics. Spartam Books, New York (1992)

[50] Ritter, G. X., Sussner, P. : An Introduction to Morphological Neural Networks. in Proceedings of the 13th International Conference on Pattern Recognition, vol. IV, Track D (1996) 709-717

[51] Argüelles, A.J., Yáñez, C., Díaz-de-León Santiago, J.L, Camacho, O.: Pattern recognition and classification using weightless neural networks and Steinbuch Lernmatrix. en: Proc. Optics & Photonics Conference 5916 Mathematical Methods in Pattern and Image Analysis, SPIE , San Diego, CA.(2005) pp. (59160)P1-P8. ISBN: 0-8194-5921-6, ISSN: 0277-786X

[52] Santiago-Montero, R.: Clasificador híbrido de patrones basado en la Lernmatrix de Steinbuch y el Linear Associator de Anderson-Kohonen. Tesis de Maestría en: Ciencias de la Computación CIC IPN, México (2003)

[53] Yáñez-Márquez, C.: Memorias Asociativas Basadas en Relaciones de Orden y Operadores Binarios. Tesis doctoral, Centro de Investigación en Computación, IPN, México (2002)

[54] Acevedo-Mosqueda, M.E.: Memorias Asociativas Bidireccionales Alfa-Beta. Tesis doctoral, Centro de Investigación en Computación, IPN, México (2006)

[55] Acevedo-Mosqueda, M.E., Yáñez-Márquez, C., López-Yáñez, I.: Complexity of Alpha-Beta Bidirectional Associative Memories. Lecture Notes in Computer Science (Revista internacional ISI), LNCS 4293, Springer-Verlag Berlin Heidelberg (2006) pp. 357-366. ISSN: 0302-9743

[56] Flores-Carapia, R.: Memorias asociativas Alfa-Beta basadas en el código Johnson-Möbius modificado. Tesis de Maestría en: Ciencias de la Computación. CIC IPN, México (2006)

[57] Yáñez-Márquez, C., Felipe-Riverón, E.M., López-Yáñez, I., Flores-Carapia, R.: A Novel Approach to Automatic Color Matching. Lecture Notes in Computer Science (Revista internacional ISI), LNCS 4225, Springer-Verlag, Berlin Heidelberg (2006) pp. 529-538. ISSN: 0302-9743

[58] Acevedo-Mosqueda, M.E., Yáñez-Márquez, C., López-Yáñez, I.: A New Model of BAM: Alpha-Beta Bidirectional Associative Memories. Lecture Notes in Computer Science (Revista internacional ISI), LNCS 4263, Springer-Verlag Berlin Heidelberg (2006) pp. 286-295. ISSN: 0302-9743

[59] Sánchez Garfias, F.A., Díaz-de-León Santiago, J.L.,Yáñez Márquez, C: Lernmatrix de Steinbuch: avances teóricos, Computación y Sistemas (Revista Iberoamericana de Computación incluida en el Índice CONACyT), Vol. 7, No. 3, México (2004) pp. 175-189. ISSN 1405-5546

[60] Yáñez Márquez, C., Díaz-de-León Santiago, J.L.: Memorias Asociativas Basadas en Relaciones de Orden y Operaciones Binarias. Computación y Sistemas (Revista Iberoamericana de Computación incluida en el Índice de CONACyT), Vol. 6, No. 4, México (2003) pp. 300-311. ISSN 1405-5546

[61] Acevedo-Mosqueda, M.E., Yáñez-Márquez, C., López-Yáñez, I.: Alpha-Beta Bidirectional Associative Memories. IJCIR International Journal of Computational Intelligence Research, Vol. 3, No. 1, México (2006) pp. 105-110. ISSN: 0973-1873

[62] Acevedo-Mosqueda, M.E., Yáñez-Márquez, C., López-Yáñez, I.: Alpha-Beta Bidirectional Associative Memories Based Translator. IJCSNS International Journal of Computer Science and Network Security, Vol. 6, No. 5A, México (2006) pp. 190-194. ISSN: 1738-7906

[63] Aldape-Pérez, M., Yáñez-Márquez, C., López -Leyva, L.O.: Optimized Implementation of a Pattern Classifier using Feature Set Reduction. Research in Computing Science, Vol. 24, Special issue: Control, Virtual Instrumentation and Digital Systems, IPN México (2006) pp. 11-20. ISSN 1870-4069

[64] Sánchez Garfias, F.A., Díaz-de-León Santiago, J.L., Yáñez Márquez, C.: New Results on the Lernmatrix Properties. Research on Computing Science Series, Vol. 10, IPN, México (2004) pp. 91-102. ISSN 1665-9899

[65] Román-Godínez, I., López-Yáñez, I., Yáñez-Márquez, C.: A New Classifier Based on Associative Memories. IEEE Computer Society, Proc. 15th International Conference on Computing, CIC México (2006) pp. 55-59. ISBN: 0-7695-2708-6

[66] Aldape Pérez, M., Yáñez Márquez, C., López Leyva L.O.: Reducción del espacio de rasgos para el diseño de Clasificadores de Patrones Optimizados. Artículo Id EO-014 en Proc. del 9o. Congreso Nacional de Ingeniería Electromecánica y de Sistemas, ESIME IPN, México (2006) pp. 64-69. ISBN: 970-36-0355-6

[67] López-Yáñez I.: Clasificador automático de alto desempeõ. Tesis de Maestría en Ciencias de la Computación. CIC IPN, México (2007)

[68] Sánchez-Garfias, F.A., Díaz-de-León Santiago, J.L, Yáñez-Márquez, C.: A new theoretical framework for the Steinbuch's Lernmatrix. en: Proc. Optics & Photonics, Conference 5916 Mathematical Methods in Pattern and Image Analysis, SPIE, San Diego, CA (2005). pp. (59160)N1-N9. ISBN: 0-8194-5921-6, ISSN: 0277-786X

[69] Hassoun, M.H.(ed.): Associative Neural Memories. Oxford University Press, New York (1993)

[70] Kohonen, T.: Self-Organization and Associative Memory. Springer-Verlag, Berlin (1989)

[71] Sossa, H., Barrón, R., Vázquez, R.: New Associative Memories to Recall Real-Valued Patterns. CIARP 2004, LNCS 3287 (2004) 195-202

[72] Díaz-de-León, J.L., Yáñez-Márquez, C.: Memorias asociativas con respuesta perfecta y capacidad infinita. Memoria del TAINA'99, México, D.F. (1999) 23-38

[73] González, R. C. , Woods, R. E.: Digital Image Processing.Prentice Hall,USA (2001)

[74] Boser, B.E., Guyon, I.M., Vapnik, V.N.: A training algorithm for optimal margin classifier. In: D. Haussler (ed.), 5th Annual ACM warshop on COLT, Pittsburgh, PA. ACM Press (1992) 144–152

[75] Cortes, C.,Vapnik V.: Support Vector Network. Machine Learning, 20 (1995) 273 – 297

[76] Yap, C.W., Xue, Y., Chen, Y.Z.: Application of Support Vector Machines to In Silico Prediction of Cytochrome P450 Enzyme Substrates and Inhibitors. Current Topics in Medicinal Chemistry, Vol. 6, Number 15. Bentham Science Publishers (2006) 1593-1607

[77] Pochet, N.L.M.M., Suykens, J.A.K.: Support vector machines versus logistic regression: improving prospective performance in clinical decision-making. Ultrasound in Obstetrics and Gynecology, vol. 27, number 6. John Wiley & Sons, Ltd. (2006) 607-608

[78] Wu, K., Yap, K.-H.: Fuzzy SVM for content-based image retrieval: a pseudo-label support vector machine framework. IEEE Computational Intelligence Magazine, Vol. 1, Issue 2 (2006) 10-16

[79] Bahamonde, A., Díez, J., Quevedo, J.R., Luaces, O., del Coz, J.J.: How to learn consumer preferences from the analysis of sensory data by means of support vector machines (SVM). Trends in Food Science and Technology, Vol. 18, Issue 1. Elsevier BV. ISSN 0924-2244 (2007) 20-28

[80] Brier, G.W. : Verification of forecasts expressed in terms of probabilities. Monthly Weather Review (1950) 78,1–3

[81] Kononenko, I.:Estimating attributes: analysis and extensions of RELIEF. Proc European Conference on Machine Learning (ECML), 171–182

[82] Hart, P.: The condensed nearest neighbor rule (Corresp.). IEEE Transactions on Information Theory, vol. 14, issue 3. ISSN 0018-9448 (1968) 515- 516

[83] Djouadi, A., Bouktache, E.: A fast algorithm for the nearest-neighbor classifier. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 19, issue 3. ISSN 0162-8828 (1997) 277-282

[84] Zhang, B., Srihari, S.N.: A Fast Algorithm for Finding k-Nearest Neighbors with Non-metric Dissimilarity. Proc. of the Eighth International Workshop on Frontiers in Handwriting Recognition, 2002. IEEE, ISBN 0-7695-1692-0 (2002) 13-18

[85] Steinbuch, K.: Die Lernmatrix. Kybernetik, vol. 1, num. 1 (1961) 36-45

[86] Salgado-Ramírez, J.C.: Estudio estadístico comparativo entre Memorias Asociativas Clásicas, Memorias Morfológicas y Memorias Alfa-Beta para el caso binario. Tesis de Maestría en :ClC IPN, México (2005)

[87] Yáñez-Márquez, C., Cruz-Meza, M.E., Sánchez-Garfias, F.A., López-Yáñez, I.: Using Alpha-beta Associative Memories to Learn and Recall RGB Images. Submitted to the Fourth International Symposium on Neural Networks ISNN (2007)

[88] Simpson, P. K.: Artificial Neural Systems. Pergamon Press, New York (1990)

[89] Vapnik , Lerner: Patern recognition using generalized pertrait method. Automation and Remote Control, (1963) 24

[90] Vapnik,V.,Chervonenkis, A.: A note on one class of Perceptrons. Automation and Remote Control, (1964) 25

[91] Drucker,H., Burges,C. J. C.,Kaufman, L., Smola,A.,Vapnik, V.: Support vector regression machines. In: M. Mozer, M. Jordan, T. Petsche(eds): Advances in Neural Information. Processing Systems 9, MA, MIT Press, Cambridge (1997) 155-161

[92] Burges,C. J. C.: A Tutorial on Support Vector Machines for Pattern Recognition. Knowledge Discovery and Data Mining, 2(2), 1998

[93] Yann LeCun, Courant Institute, NYU Corinna Cortes, Google Labs, New York The MNIST database of handwritten digits http://yann.lecun.com/exdb/mnist/

[94] LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-Based Learning Applied to Document Recognition, In Proceedings of the IEEE 86(11), 2278-2324 (1998)

[95] Simard, P.Y., Steinkraus, D., Platt, J.: Best Practice for Convolutional Neural Networks Applied to Visual Document Analysis. In International Conference on Document Analysis and Recognition (ICDAR), pp. 958-962. IEEE Computer Society, Los Alamitos (2003)

[96] Decoste, D., Scholkopf, B.: Training Invariant Support Vector Machines. Machine Learning 46, 161-190 (2002)

[97] Salakhutdinov, R.R., Hinton, G.E.: Learning a Nonlinear Embedding by Preserving Class Neighbourhood Structure. In AI and Statistics, Puerto Rico (2007)

[98] Ranzato, M.A., Huang, F.J., Boureau, Y.L., LeCun, Y.: Unsupervised Learning of Invariant Feature Hierarchies with Applications to Object Recognition, In Proc. Computer Vision and Pattern Recognition Conference (CVPR'07), IEEE Press (2007)

[99] Ranzato, M.A., Poultney, C., Chopra, S., LeCun, Y.: Efficient Learning of Sparse Representations with an Energy-Based Model, In J. Platt et al. (Eds), Advances in Neural Information Processing Systems (NIPS 2006), MIT Press (2006)

[100] S. Cook, R. Reckhow: Time-bounded random access machines. Journal of Computer Systems Science 7, 354-375, 1963.

[101] Weisstein, Eric W. "Zero Vector." From MathWorld–A Wolfram Web Resource. http://mathworld.wolfram.com/ZeroVector.html

[102] Weisstein, Eric W. "Factorial." From MathWorld–A Wolfram Web Resource. http://mathworld.wolfram.com/Factorial.html

[103] Mano, Morris. Diseño Digital, Ad. Prentice-Hall (2001)

[104] Otsu, N. "A threshold selection method from gray-level histograms". IEEE Trans. Sys., Man., Cyber. 9 (1976) 62–66

[105] Newman, D.J., Hettich, S., Blake, C.L., Merz, C.J.: UCI Repository of machine learning databases. Irvine, CA: University of California, Department of Information and Computer Science. (1998) Available at: http://archive.ics.uci.edu/ml/

[106] Ou, G., Murphey, Y.L.: Multi-Class Pattern Classification Using Neural Networks. Pattern Recognition Vol. 40(1). Pattern Recognition Society. Elsevier Ltd. (2007) 4–18

[107] Fletcher, P., Hughes, H. & Wayne C., Foubdation of Discrete Mathematics, PWS-Kent Publishing Company (1991)

Apéndice A

Diagrama de flujo del algoritmo

Figura A.1 Diagrama de flujo para la fase de aprendizaje

Figura A.2 Diagrama de flujo para la fase recuperación - primera parte

Figura A.3 Diagrama de flujo para la fase recuperación - segunda parte

Figura A.4 Diagrama de flujo para la fase recuperación - tercera parte

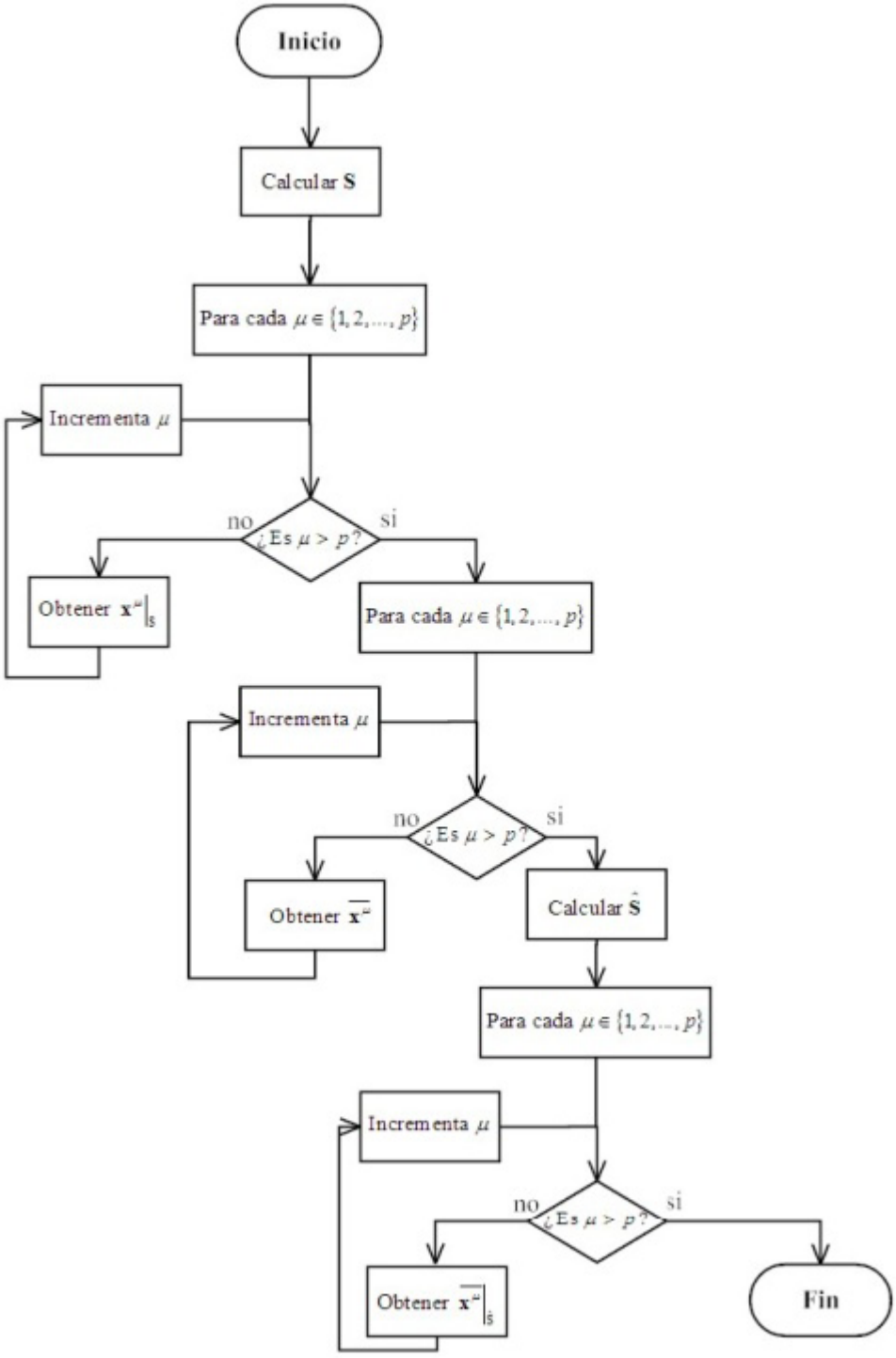


Figura A.1: Fase de Aprendizaje de las máquinas asociativas Alfa-Beta con soporte vectorial

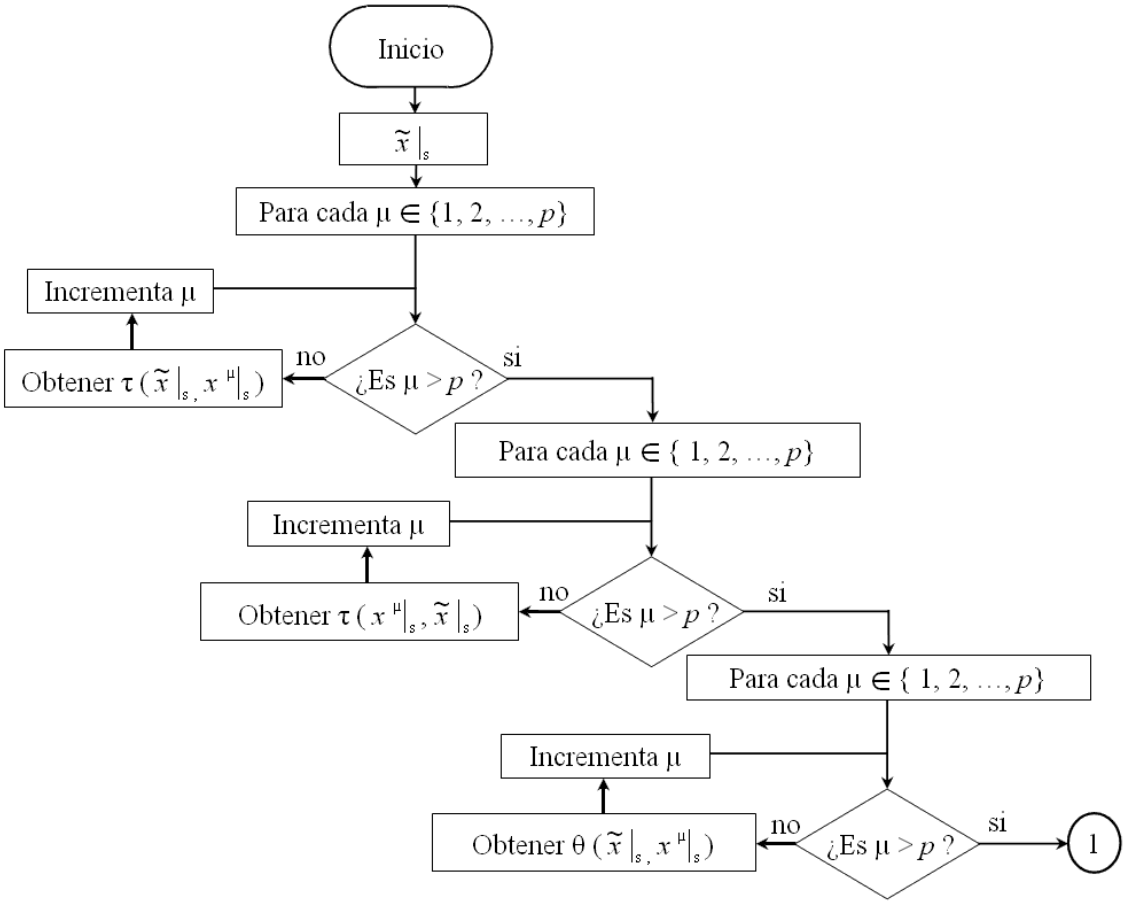


Figura A.2: Fase de Recuperación de las máquinas asociativas Alfa-Beta con soporte vectorial, parte 1

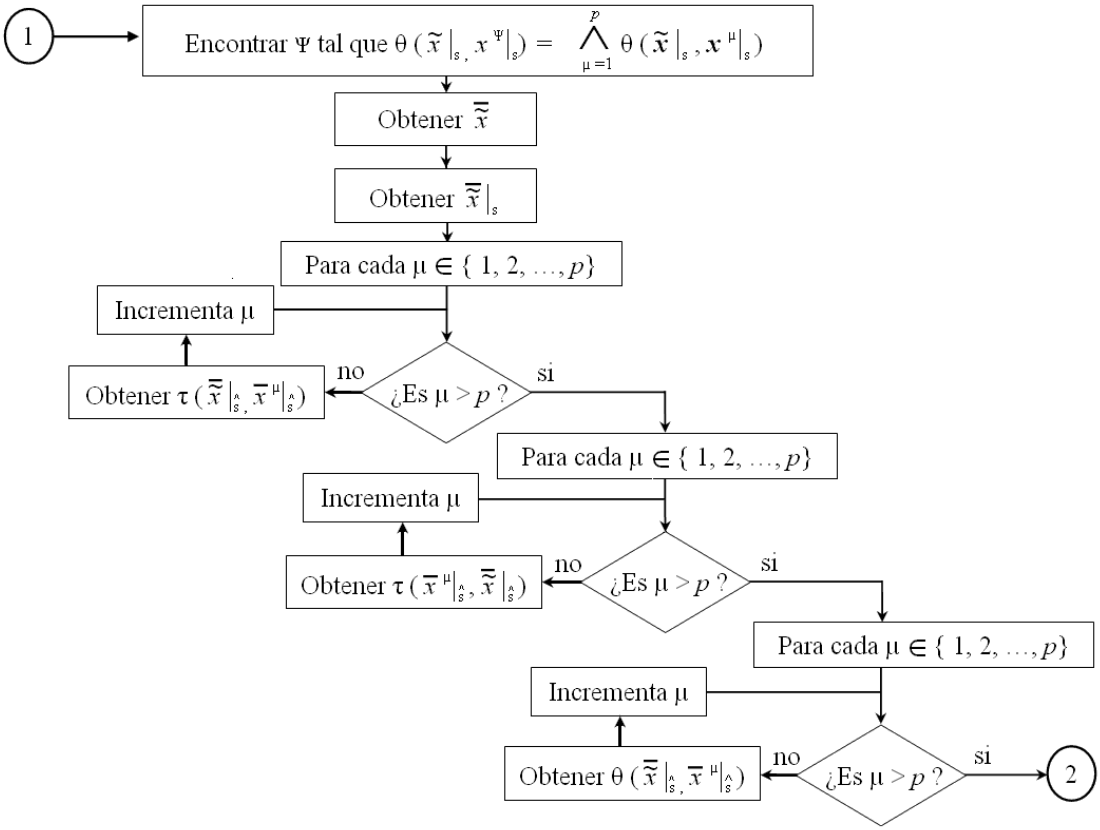


Figura A.3: Fase de Recuperación de las máquinas asociativas Alfa-Beta con soporte vectorial, parte 2

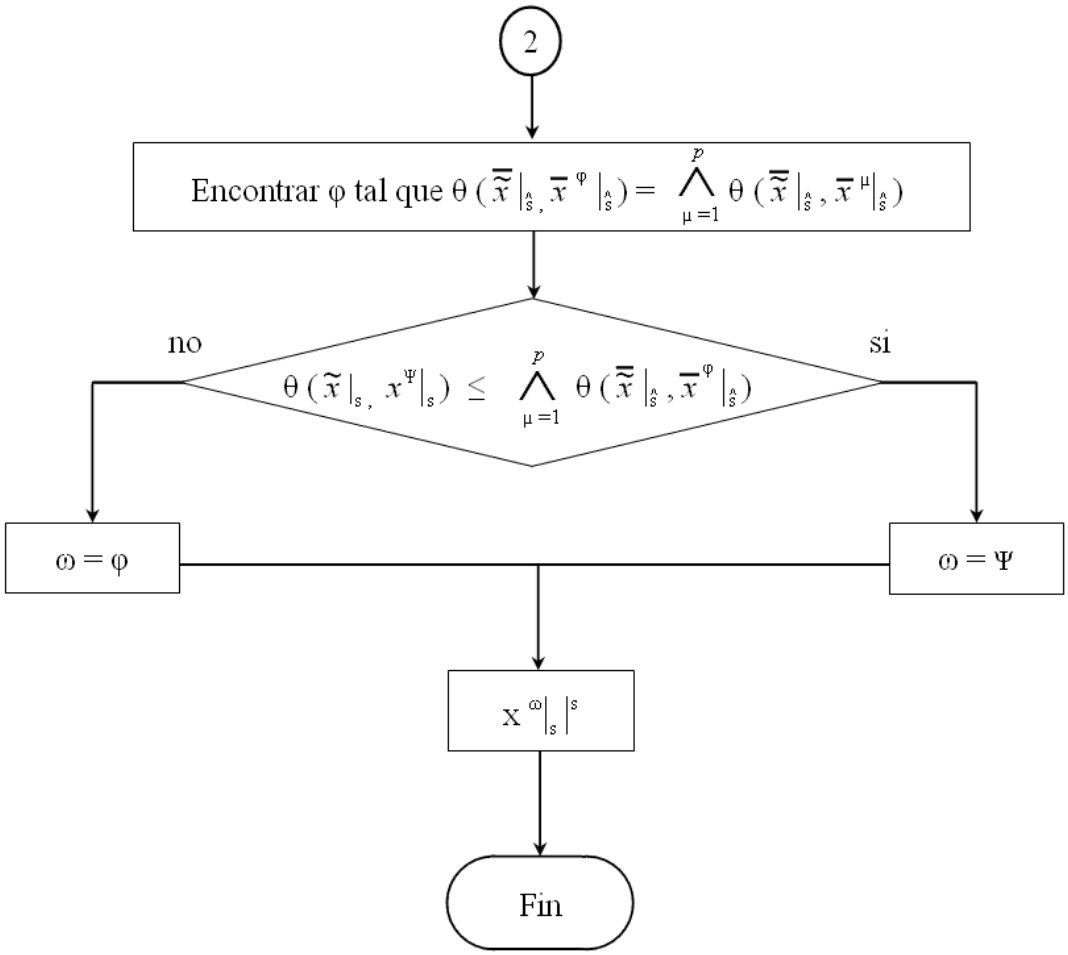


Figura A.4: Fase de Recuperación de las máquinas asociativas Alfa-Beta con soporte vectorial, parte 3