

UNIVERSIDAD DE CONCEPCIÓN
DEPARTAMENTO DE INGENIERÍA INFORMÁTICA

ESTUDIO ESPACIO-TEMPORAL DE PATRONES EN EL
MODELAMIENTO DISCRETO DE LA DINÁMICA DELICTUAL

Tesis de Magíster de Ciencias de la Computación

Bruno Francisco Karelovic Burotto

Profesor guía
Dr. Julio Aracena

Agosto 2011
CONCEPCIÓN-CHILE

Índice

1	Introducción	1
1.1	Autómatas Celulares	3
1.1.1	Sistemas complejos	4
1.1.2	Historia de los AC	4
1.1.3	Definición	4
1.1.4	Ejemplo	5
1.2	Redes Neuronales Artificiales	7
1.2.1	Perceptrón	7
1.2.2	Perceptrón multicapa	8
2	Patrones espaciales	12
2.1	Datos	12
2.2	Configuraciones prohibidas	15
2.3	Distribución de las configuraciones	17
2.4	Diferenciar prohibidas	18
3	Modelamiento discreto de la dinámica delictual	20
3.1	Dinámica observada	20
3.2	Autómata Celular	21
3.3	Redes Neuronales Artificiales	23
3.4	Desempeño del modelo	28
3.5	Otros modelos	29
3.6	Dinámica y control en Santiago	36
3.7	Dinámica y control en Ñuñoa	40
4	Herramientas computacionales	45
4.1	Matlab	45
4.2	PHP	46

5 Conclusiones	48
A Dinámica observada y del autómata para Santiago	50
B Problema combinatorial de trayectorias	52
Bibliografía	55



Capítulo 1

Introducción

Actualmente, el crimen es una de las principales preocupaciones de la población, pues está relacionada directamente con la calidad de vida. Se presenta en casi todas las grandes ciudades pero no de manera uniforme. De hecho, los crímenes se agrupan concentradamente en ciertas zonas llamadas *hotspots*. Estudios revelan que cuando las acciones policiales están dirigidas a estos hotspots, disminuye el número de crímenes junto a las llamadas a la policía [4].

La mayoría de los trabajos analíticos relacionados con el crimen han estado focalizados en determinar los hotspots de una zona dada, con objeto de predecir la ocurrencia de delitos con probabilidad alta [20, 21]. Por otro lado, también hay muchos trabajos que buscan explicar estos hotspots con un enfoque social y econométrico [8, 19].

Más recientemente, se han desarrollado trabajos que buscan describir los patrones, ya no sólo estáticos, si no espacio-temporales de los hotspots con un enfoque continuo, utilizando EDP [3, 11, 24]. Por ejemplo en [22], Short, Bertozzi y Brantingham analizan la formación y dinámica de los hotspots por medio de un modelo basado en las EDP de reacción y difusión.

Por otro lado, la actividad delictual de una zona geográfica dada puede ser visto como la resultante de una componente local propia de una región al interior de la zona considerada, más una tendencia global. Existen trabajos relacionados con el intento de separar ambas componentes del nivel delictual observado en el tiempo [2].

Uno de los fenómenos del crimen, que explica en parte la presencia de patrones espacio-temporales globales, es el llamado *near repeat* [23] (Figura 1.1). Esto es, cuando ocurre un crimen, en las zonas cercanas aumenta el riesgo de ser atacadas. En particular, el efecto de las *ventanas rotas* [25] representa el hecho que si en un punto ocurre un crimen, entonces aumentará la posibilidad que ese punto sufra un nuevo daño más adelante.

Estas propiedades han sido utilizadas para modelar la dinámica delictual. Así, en [24], Short et al. muestran un modelo que trata a los criminales como agentes que se desplazan en una grilla, y se mueven según la atractividad de la zona que depende por un lado de un factor fijo y por otro de la historia de crímenes en los alrededores.

Dado que el crimen tiene una fuerte componente local e histórica, como muestran los fenómenos de near repeat y ventanas rotas, además de poseer una fuerte no-linealidad [22], los autómatas celulares aparecen como un modelo natural para representarlos.

Los *autómatas celulares* (AC) son una representación simple y útil para estudiar la dinámica de los sistemas complejos. Estos evolucionan con reglas locales de las cuales pueden emerger comportamientos globales. Los AC han sido usados para simular procesos espaciotemporales desde los años 90 [9]. Casi todas estas aplicaciones son fenómenos visibles, en particular ha sido utilizado para fenómenos sociales, como el trabajo de Dabaghinian et al. [7], que modela la migración residencial y tangencialmente el crimen también.

Jun Liang en 2001 [12], simuló el crimen y los patrones delictuales con un autómata celular enfocándose en analizar particularmente la interacción entre delincuentes y víctimas. Define el concepto de *tensión* como el impacto de los crímenes en las víctimas y su reacción a ellos. Así, con el AC modela la difusión de la tensión en el espacio y tiempo.

Por otro lado, las redes neuronales artificiales (RN) son un modelo matemático inspirado en las redes neuronales biológicas, usado principalmente para encontrar patrones en datos asociados a un fenómeno altamente no lineal [9].

El modelamiento de la actividad delictual por autómatas celulares y redes neuronales tiene sus inicios en un trabajo realizado por Andreas Olligschlaeger y Wilpen Gorr [18], quienes buscan predecir la aparición de focos de tráfico de drogas. Para ello los autores proponen un modelo híbrido en base a autómatas celulares y redes neuronales artificiales. El procedimiento consiste en modelar los datos tal como si de un autómata celular se tratase, dividiendo el plano de la ciudad en celdas y una vez definida la partición y los datos que corresponden a cada celda, redes neuronales son entrenadas para aprender el comportamiento o reglas locales que rigen la dinámica del modelo. El resultado no necesariamente es un AC, pues las funciones locales no son las mismas en todas las celdas.

En este trabajo, definimos como modelo matemático de la actividad delictual un autómata celular con la ayuda de redes neuronales para la extracción de patrones espacio-temporales observables en la componente global de la dinámica del crimen en una zona geográfica dada. En particular, para la actividad delictual asociada a hurtos en Santiago durante 2001 a 2004 y Ñuñoa entre 2007 y 2010.

A través de la definición del modelo se estudiará la influencia de la vecindad y la historia en la capacidad de predecir el delito en una zona dada.

Se buscará caracterizar la dinámica observada, determinando sus propiedades, como la influencia de la difusión, concentración y saturación del crimen a través del tiempo. También conocer algunos puntos fijos y otros ciclos límites. Además, utilizar el modelo como herramienta de control, contrastando las variaciones en la evolución de una configuración real y simular la dinámica delictual en escenarios no vistos anteriormente.

Por otro lado, se estudiará la emergencia de configuraciones globales a partir de interacciones locales. En particular, entender la presencia o ausencia de algunas configuraciones en la dinámica observada.

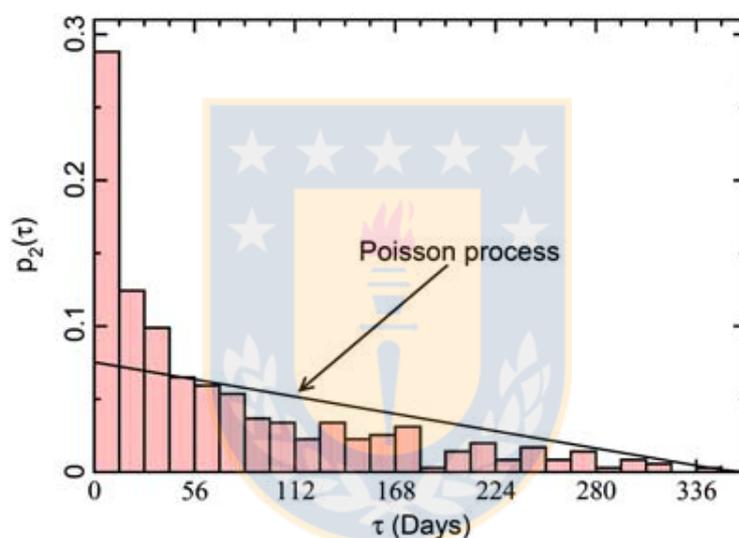


Figura 1.1: Probabilidad de observar un lapso de τ días entre eventos en residencias entradas a robar. Imágen de [23]

1.1 Autómatas Celulares

Los Autómatas Celulares (AC) son una de las representaciones más sencillas de los sistemas complejos, y han mostrado ser muy útiles en el modelamiento de la dinámica de muchos sistemas complejos, como la física de fluidos, ecología, tráfico, economía, sistemas dinámicos moleculares, entre otros.

Debido a su simplicidad, los AC son mecanismos conceptuales útiles para estudiar formaciones de patrones generales.

1.1.1 Sistemas complejos

Se pueden ver numerosos ejemplos de sistemas complejos en la naturaleza, desde la turbulencia de los fluidos hasta la complejidad de organismos vivos, todos estos fenómenos comparten la siguiente propiedad: ellos consisten en la unión de muchas partes que interactúan entre ellas. A partir de ello se logra un comportamiento complejo que *emerge*.

Un clásico ejemplo de sistema complejo es el cerebro humano, que consiste en unas 10^{10} neuronas con $10^3 - 10^4$ conexiones por neurona. Siendo quizás el sistema complejo más complejo. Así, de alguna manera, de una cooperación dinámica en esta gran red interconectada de neuronas *emerge* una estructura suficientemente compleja y coherente como para poder estar estudiando este mismo comportamiento.

1.1.2 Historia de los AC

En los años 1940, mientras Stanislaw Ulam estudiaba el crecimiento de los cristales utilizando un modelo discreto, John von Neumann buscaba un modelo sencillo de autoreplicación biológica cuyo diseño inicial era encontrar un robot que construyese otro robot. Ulam sugirió que desarrollara la idea con una abstracción matemática como él lo estaba haciendo para estudiar el crecimiento de cristales, para así evitar los problemas prácticos. Esto resultó en el primer AC.

Ya en los años 1970, John Conway inventó un AC bidimensional llamado “Juego de la Vida”, siendo quizás el más conocido actualmente [10].

1.1.3 Definición

Aunque hay una gran variedad de AC para casos particulares, la mayoría de ellos comparte estas cinco características genéricas:

- **Grilla discreta de celdas:** El sistema consiste en una grilla infinita uni-, bi- o tridimensional de celdas.

- **Estados discretos:** Cada celda tiene un número finito de posibles estados discretos.

- **Interacciones locales:** Cada celda interactúa sólo con las celdas que se encuentran en su vecindad: un conjunto finito de celdas en su cercanía.

- **Dinámica discreta:** En cada unidad discreta de tiempo, cada celda actualiza su valor de acuerdo a una *regla de transición* tomando los estados de los valores de las celdas vecinas y de ella misma.

- **Homogeneidad:** Todas las celdas son equivalentes, es decir, la regla de transición se aplica de forma homogénea a todas las celdas. Y la vecindad es la misma para cada celda, es decir, es invariante frente a traslación.

1.1.4 Ejemplo

Quizás el AC más conocido es el *Juego de la Vida* propuesto por John H. Conway y popularizado por Martin Gardner en “Mathematical Games” a comienzos de los años 70.

Este autómata utiliza la vecindad de Moore (Figura 1.2), con dos estados: celdas vivas, a las cuales se le asigna el valor 1, y celdas muertas, cuyo valor es 0. La regla de transición que se aplica repetidamente en tiempos discretos, se define como sigue:

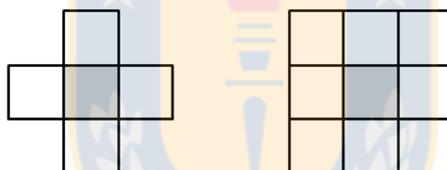


Figura 1.2: Ejemplos de vecindades en dos dimensiones. La primera corresponde a la vecindad de von Neumann, la segunda a la de Moore.

Nacimiento: Una celda pasa de estado 0 a estado 1 si en sus vecinos hay exactamente 3 celdas vivas.

Muerte: Una celda cambia de estado 1 a 0 si hay 0 ó 1 celda viva, es decir, muere por soledad. O bien, si hay 4 ó más celdas vivas, es decir, muere por sobrepoblación.

Sobrevive: Una celda viva se mantiene en ese estado si tiene 2 ó 3 vecinos vivos.

A partir de estas sencillas reglas, emerge una dinámica compleja (Figura 1.3), dentro

de la cual existen numerosos elementos. Entre los cuales se encuentran los “planeadores” (Figura 1.4), estructuras que se mueven a través del espacio; “trenes”, que aumentan constantemente; figuras estáticas; “matuhsalens”, cuyo transiente es muy largo a partir de una semilla pequeña, etcétera.

En efecto, este AC es una máquina Universal de Turing , es decir, todo lo que se puede computar algorítmicamente, se puede computar en el juego de la vida.

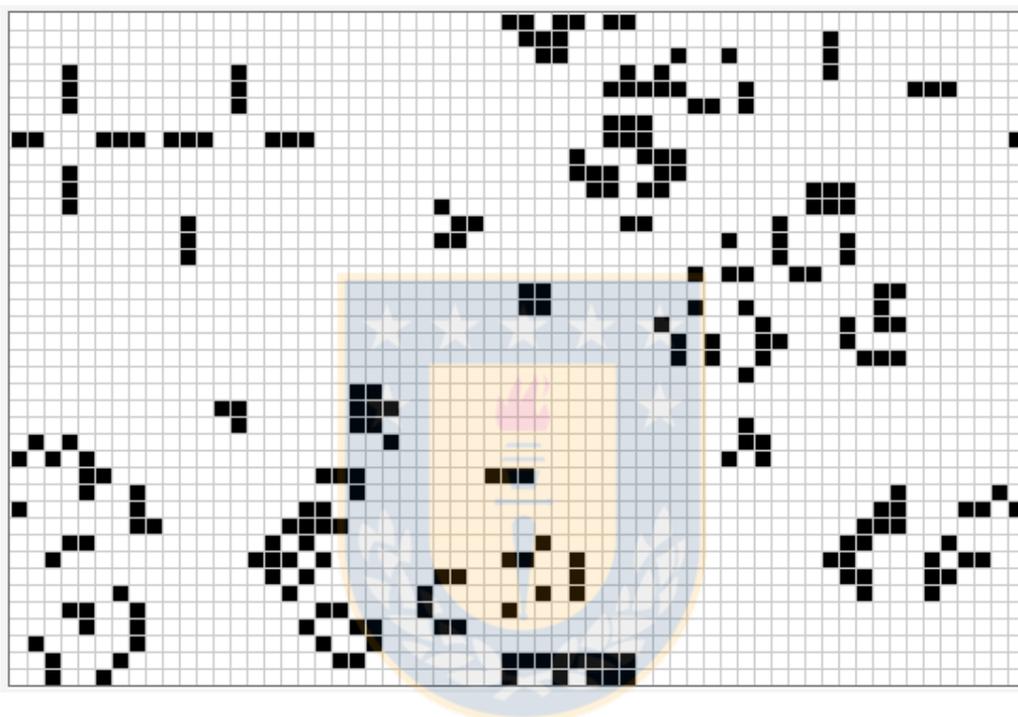


Figura 1.3: Ejemplo de una unidad de tiempo en el juego de la vida.

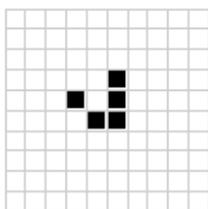


Figura 1.4: “planeador”: es un patrón que viaja a través de la grilla diagonalmente.

1.2 Redes Neuronales Artificiales

Las redes neuronales artificiales son un modelo matemático inspirado en la estructura y funcionamiento de las redes neuronales biológicas, ya que consisten en un grupo de neuronas interconectadas que procesan la información utilizando un enfoque conexionista.

Las RN son apropiadas para aplicaciones en las que no se conoce a priori el modelo pero se tiene un conjunto de ejemplos. Además, son robustas ante el ruido.

Por esta razón, son utilizadas para problemas de clasificación y reconocimiento de patrones, entre otros.

1.2.1 Perceptrón

Rosenblatt en 1958 [16] introdujo un modelo sencillo de red neuronal llamado perceptrón, que es un clasificador binario que utiliza como entrada un vector n -dimensional de valores reales x , es decir

$$f(x) = \begin{cases} 1 & \text{si } w \cdot x + b \geq 0 \\ 0 & \text{si } w \cdot x + b < 0 \end{cases} \quad (1.1)$$

donde w es un vector de pesos y b es el *umbral*, una constante que no depende de los valores de entrada.

El valor de $f(x)$ se utiliza para clasificar x como instancia positiva o negativa. Espacialmente, puede verse a b como la posición que debe superar $w \cdot x$ para que x sea clasificado positivo.

Para encontrar la regla apropiada, es decir, los valores de w y b , se presentan cada ejemplo x^i al perceptrón y según su salida $y(x^i)$ y salida esperada z^i en $\{0, 1\}$ se actualizan los pesos de la siguiente manera

$$w_j(t+1) = w_j(t) + \eta(y_j^i - z_j^i)x_j^i$$

donde $0 \leq \eta \leq 1$ es la *constante de aprendizaje* que controla la tasa de aprendizaje. Con esa regla, los pesos permanecen inalterados si la salida del perceptrón es igual a la salida esperada.

Dado un conjunto de entradas clasificadas en dos grupos, el perceptrón podrá separarlos si y sólo si estos conjuntos son *linealmente separables* [14], es decir, se puede encontrar un hiperplano (o línea en el caso de dos dimensiones y plano en el de 3) que deje a un lado todos los puntos de un categoría y al otro, el conjunto restante.

Las implicancias de este resultado hacen que un problema sencillo llamado XOR (Tabla 1.1), no pueda ser aprendido por el perceptrón

x_1	x_2	$x_1 \text{ XOR } x_2$
0	0	0
0	1	1
1	0	1
1	1	0

Tabla 1.1: Función XOR.

Un modelo apropiado de perceptrón tendría dos neuronas, correspondientes a x_1 y x_2 , tal que $y = x_1 \text{ XOR } x_2$ para algunos w_1 , w_2 y b . Se puede observar que tal problema es imposible. Es decir, el perceptrón no puede aprender la regla XOR.

Este resultado de Minsky en 1969 desmotivó la investigación en el área de las redes neuronales, disminuyendo considerablemente las publicaciones relacionadas, dejando sólo a unos pocos investigadores en ello, sin embargo, el perceptrón multicapa solucionaría esta fuerte limitante.

1.2.2 Perceptrón multicapa

Para modelar problemas cuyos puntos no sean linealmente separables, entre ellos el problema *XOR*, se agregaron capas ocultas y así se crea el *perceptrón multicapa*, que es una generalización del perceptrón. Más aún, Cybenko demostró en 1986 [6] que una red neuronal feedforward con una única capa oculta que contenga un número finito de neuronas y función de transición sigmoideal, es un aproximador universal para cualquier función real. Además, se sabe que para funciones discretas, una red neuronal puede simularla exactamente con dos capas ocultas pero un gran número de neuronas [1].

Sin embargo, en el caso de las funciones reales el resultado es sólo de existencia y no se sabe cómo construir la red. Por otro lado, en las funciones discretas se necesita un gran número de neuronas para lograr que la red se comporte igual que la función. Por estas razones se ocupa un número distintos de capas y neuronas, ya que el número de capas aumenta las posibilidades de no linealidad.

Las redes neuronales que utilizaremos nosotros corresponde a la arquitectura *feedforward*, es decir, las conexiones entre las neuronas son siempre hacia las capas siguientes de neuronas, nunca hacia la misma capa ni hacia atrás. Más aun, serán estrictamente feedforward, porque las conexiones sólo irán a la capa inmediatamente después. En la Figura 1.5

se ilustra la arquitectura de un perceptrón multicapa feedforward.

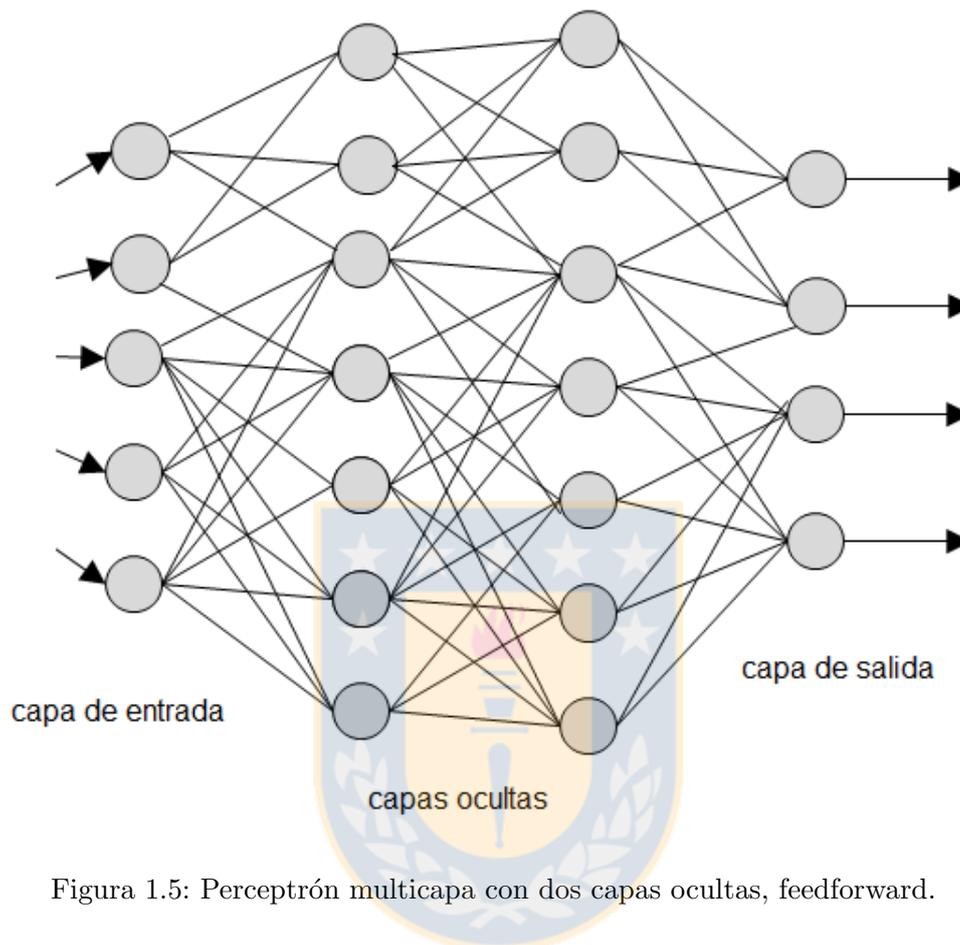


Figura 1.5: Perceptrón multicapa con dos capas ocultas, feedforward.

Usualmente, la salida de cada neurona no es lineal, si no que antes se aplica una *función de activación*, la que utilizamos nosotros es la función sigmoideal, donde, x es la salida de la neurona y $f(x)$ es el valor que recibirán las neuronas conectadas hacia adelante de ella.

$$f(x) = \frac{1}{1 + e^{-\alpha x}}$$

con α un parámetro para ajustar la pendiente (Figura 1.6).

Para entrenar estas redes, se utiliza la regla de aprendizaje *Backpropagation* para perceptrones multicapas, creado por Rumelhart y McClelland [17] en 1986. Al igual que en el caso del perceptrón, el objetivo fundamental es, a partir de un grupo de datos, minimizar el error cuadrático medio entre el valor entregado por la red y la salida esperada. Ya que esta regla requiere conocer la salida deseada, se dice que es *aprendizaje supervisado*.

Por ejemplo, si el problema es aprender las 27 letras del alfabeto, la entrada sería un conjunto de ejemplos de letras escritas a mano en una grilla de $M \times N$ donde cada pixel

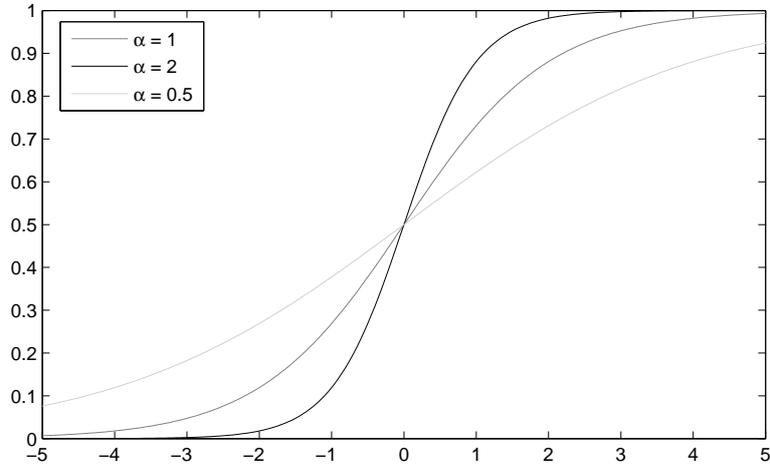


Figura 1.6: Función sigmoideal para distintos valores de α .

corresponda a una neurona y su valor sea 0 si el pixel es blanco y 1 si es negro. La capa de salida corresponde a 27 neuronas, una por cada letra. Así, una perfecta asociación entre entradas y salidas sería que para cada ejemplo de letra de entrada, todas las neuronas de salida sean cero, excepto las que corresponden a la letra de entrada, que valdría 1.

En cuanto a las capas ocultas, pueden haber varias entre las capas de entrada y salida con un valor arbitrario de neuronas cada una. Ya que no existen teoremas que indiquen la cantidad óptima de neuronas en las capas ocultas, se utilizan heurísticas.

Consideremos una red con una capa intermedia. Sea σ_k^p el valor de la salida de la k -ésima neurona de entrada correspondiente al p -ésimo ejemplo. Sea h_j el valor de salida de las neuronas de ocultas y z_i el valor de las neuronas de salida. $\{w_{jk}\}$ es el conjunto de pesos de las conexiones entre la capa de entrada y la oculta, mientras $\{w'_{ik}\}$ son los pesos entre la capa oculta y la de salida. Finalmente, sea d_i^p el valor esperado para la i -ésima neurona del ejemplo p . Es decir, el conjunto de entrenamiento consiste en el par $\{\sigma_k^p, d_k^p\}$.

Así, el cambio de los pesos de la capa oculta a la de salida es

$$\delta w'_{ij} = -\eta \sum_p (d_i^p - z_i^p) f'_\alpha(\mu_i^p) h_j^p.$$

Con $0 < \eta \leq 1$ una constante de aprendizaje y donde $f_\alpha(x)$ es la función sigmoideal con parámetro α para controlar la pendiente de la curva:

$$f_\alpha(x) = \frac{1}{1 + e^{-\alpha x}}.$$

Y μ_i^p es lo que recibe la neurona de salida i . Luego,

$$\mu_i^p = \sum_j w'_{ij} h_j^p = \sum_j w'_{ij} f_\alpha \left(\sum_k w_{jk} \sigma_k^p \right).$$

Por otro lado, para los pesos de la entrada a la capa oculta se tiene:

$$\delta w_{jk} = \eta \sum_{p,i} (d_i^p - z_i^p) f'_\alpha(\mu_i^p) w'_{ij} f'_\alpha(\mu_i^p) \sigma_k^p$$

En consecuencia, se obtiene un proceso recursivo. Para más detalle revisar, por ejemplo, el libro de Ilachinski [9].

Así, los parámetros de las redes feedforward son: la arquitectura, que en nuestro caso particular está dado sólo por el número de capas ocultas y la cantidad de neuronas de ellas; función de activación para cada neurona y algoritmo de entrenamiento.



Capítulo 2

Patrones espaciales

2.1 Datos

Se cuenta con un conjunto de delitos georreferenciados de la comisaría de Santiago entre los años 2001 y 2004 y de la comuna de Ñuñoa de 2007 a 2010. Los datos de Santiago incluyen 25000 crímenes reportados. Cada uno de ellos incluye información del tiempo, localización y naturaleza del delito. Nosotros trabajaremos sólo con los delitos tipo *hurto*, los cuales cuentan con 11500 eventos en Santiago (Figura 2.1) y 12000 en Ñuñoa (Figura 2.2). En adelante, se trabajará con ambos datos, pero por defecto será con Santiago, es decir, en caso de no mencionarse la zona tratada, se referirá a Santiago.

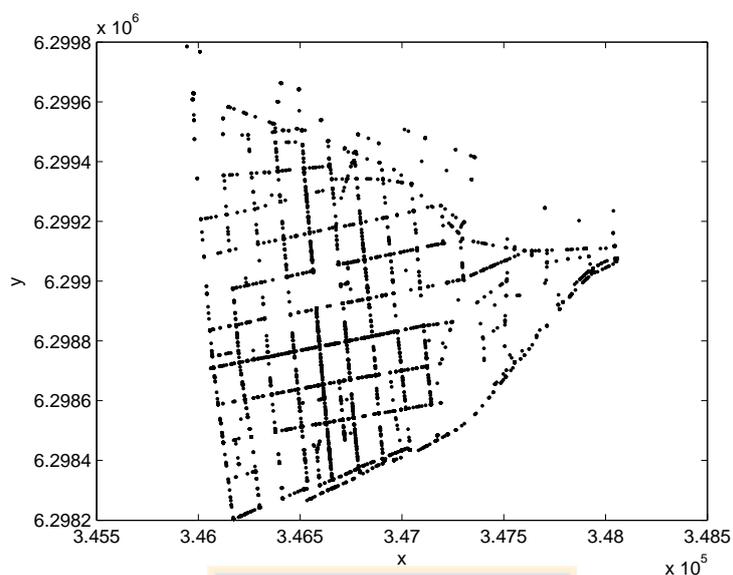


Figura 2.1: Hurtos entre los años 2001 y 2004 en la comisaría de Santiago.

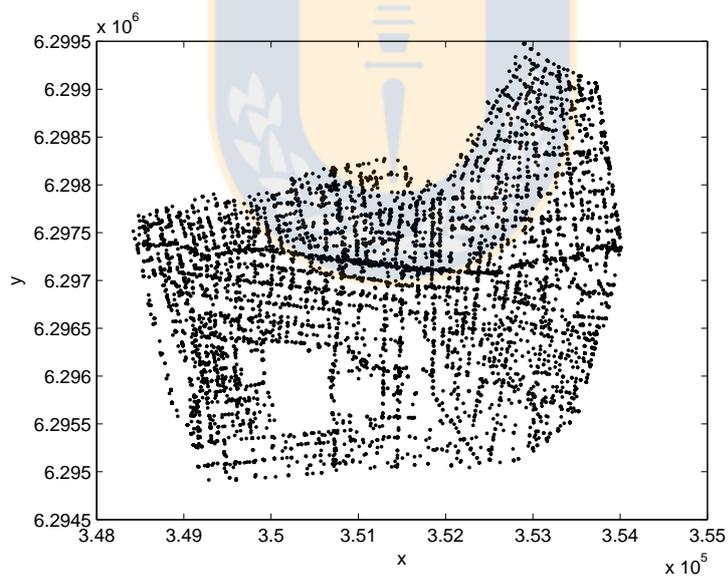


Figura 2.2: Hurtos entre los años 2007 y 2010 en la comuna de Ñuñoa.

Para tratar estos datos discretamente, se optó por construir una grilla cuadrada de $200 \text{ m} \times 200 \text{ m}$ en el caso de Santiago, de acuerdo a lo considerado por Carabineros. Y discretizar el tiempo cada una semana. Así, se obtuvo una serie de 208 configuraciones, respectivas a cada semana, donde el valor de celda indica el número de delitos que ocurrió

durante esa semana en particular (Figura 2.3).

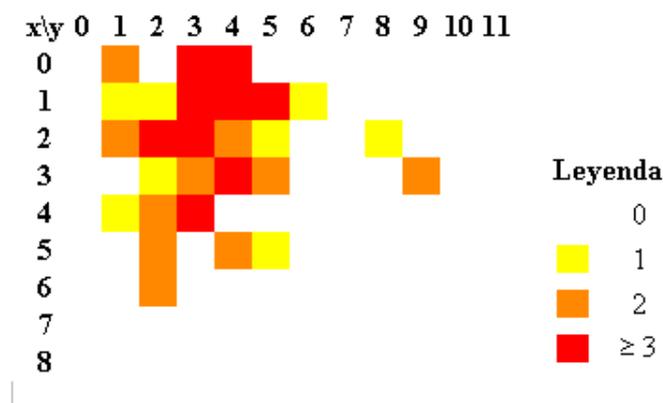


Figura 2.3: Crímenes por celda en Santiago en la semana número 20.

Como se observa en la Figura 2.4, hay celdas que no presentan crimen alguno en toda su historia, estas celdas serán excluidas y las otras les llamaremos *celdas importantes*. En la Figura 2.5 se muestran en gris.

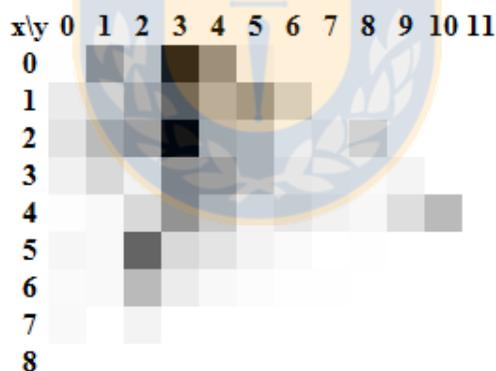


Figura 2.4: Crímenes acumulados en Santiago por celda.

Para disminuir el número de estados, el conjunto de estados se truncó a $\{0, 1, 2, \geq 3\}$, ya que además, los valores superiores a 3 representaban una pequeña fracción de las observaciones (Figura 2.6).

Para Ñuñoa se trabajó con celdas de $600m \times 600m$ pues los crímenes estaban más dispersos. En la Figura 2.7 se observa una semana de Ñuñoa con tamaño $200m \times 200m$ (A) y $600m \times 600m$ (C). En el primer caso, al estar muy dispersos los datos, un AC no podría modelar la dinámica correctamente, pues su vecindad serían normalmente cero.

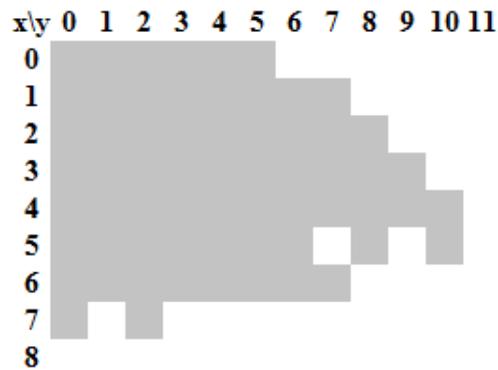


Figura 2.5: Celdas importantes de Santiago, es decir, las 63 celdas que presentan al menos un crimen en su historia.

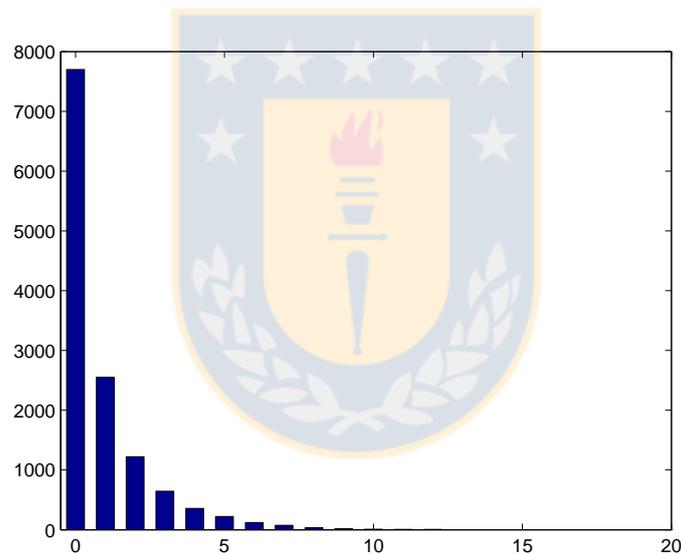


Figura 2.6: Histograma de valores de cada celda en Santiago.

Como se observó en estos mapas, Ñuñoa es más grande que Santiago pero como se comentó al comienzo, ambas tienen similar número de crímenes.

2.2 Configuraciones prohibidas

Si se observa la dinámica delictual en distintas semanas, se observa que los *hotspots* están formados por celdas con alta criminalidad rodeados por otras con cada vez menos

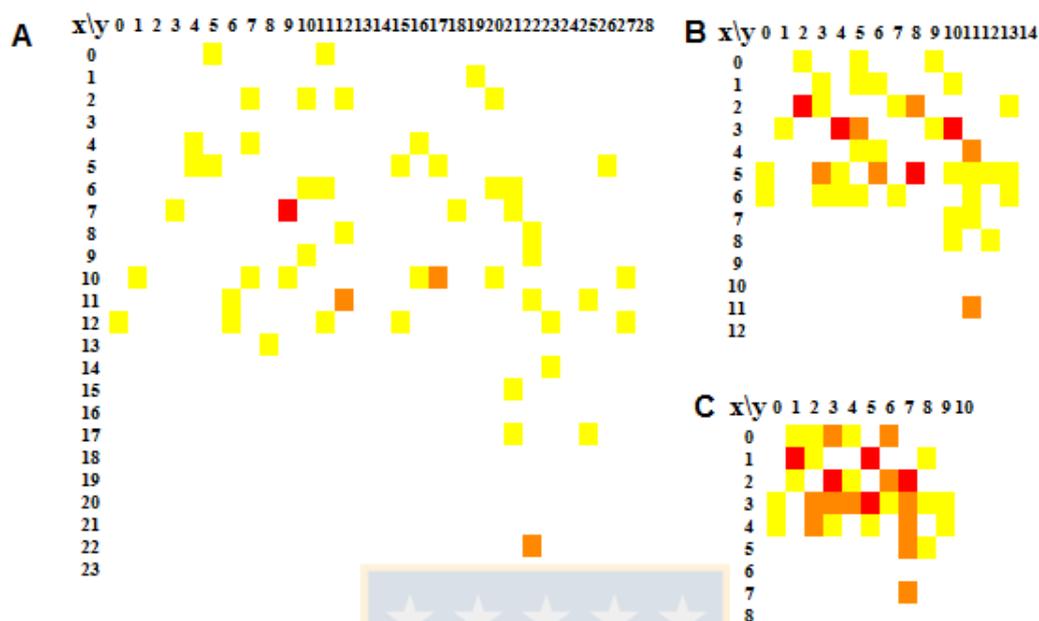


Figura 2.7: Semana 1 de Ñuñoa según tamaño de la celda. A) $200m \times 200m$. B) $400m \times 400m$. C) $600m \times 600m$.

crímen. Lo que da indicios de haber cierto patrón en la manera en que se distribuyen las configuraciones vistas en el espacio total de posibles configuraciones.

Además, considerando que se observaron 206 semanas con 88 celdas, se tiene que el máximo de configuraciones observables de 3×3 es $206 \times 88 = 18128$ configuraciones, pero sólo se observaron alrededor de 6000. Es decir, sólo un 33%. A las configuración no observadas las llamaremos *configuraciones prohibidas*.

Para confirmar esta suposición se generó una lista de configuraciones que no hayan sido observadas sin que se repitan entre ellas. El número de estas configuraciones no vistas correspondía al número de observadas. Así, a partir de estas dos listas se construyó una red neuronal con dos capas ocultas de 5 y 3 neuronas respectivamente, que fue capaz de diferenciar ambos grupos con un 77% de aciertos.

Sin embargo, teniendo en cuenta que es más común encontrar configuraciones con celdas de bajo número de crímenes, se optó por construir una segunda lista de configuraciones no vistas, pero esta vez el valor de las celdas no era escogido con igual probabilidad. Si no que proporcional a la frecuencia con que aparece cada número. Así, de todas maneras se alcanza un 66% de aciertos.

Un buen modelo debería entonces tender a entregar pocas configuraciones prohibidas,

si es capaz de tomar el comportamiento global.

2.3 Distribución de las configuraciones

Considerando la misma ventana de 3×3 , se tiene que el espacio de posibles configuraciones es $\{0, 1, 2, 3\}^9$, totalizando 262144 elementos. Aunque ya vimos que podían diferenciarse las configuraciones observadas, es de interés conocer la distribución de esas 6000 configuraciones en el espacio total.

Utilizando la norma $L1$, es decir

$$|x|_1 = \sum_{i=1}^n |x_i|$$

el diámetro del espacio es 27. Además, se encontraron 225 configuraciones que a distancia 4 no existía configuración que haya sido observada. Esto habla de 4 esferas de diámetro 8 que eventualmente podrían tener intersección no vacía. (Figura 2.8).

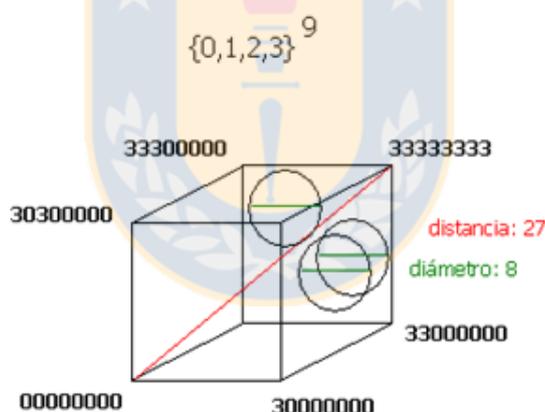


Figura 2.8: Representación del espacio de las posibles configuraciones. Hay 225 esferas de diámetro 8 que no tienen ninguna configuración observada en su interior.

En general, pueden encontrarse configuraciones que a distancia n no hayan configuraciones observadas. A este conjunto se le llamará *aisladas* A_n . Notar que A_0 coincide con el conjunto de las configuraciones no observadas. En la Tabla 2.1 se observa el tamaño de cada conjunto.

n	$ A_n $
0	256350
1	214689
2	106939
3	16526
4	225
≥ 5	0

Tabla 2.1: Configuraciones aisladas.

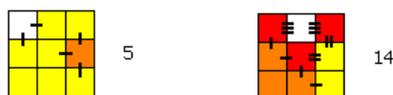
2.4 Diferenciar prohibidas

Una pregunta natural sería conocer si existe alguna característica que diferencie a ambos conjuntos (observados y no observados). Se obtuvo que una RN fue capaz de diferenciar ambos conjuntos con un 70% de aciertos. Lo que da indicios de que existe un patrón.

Sin embargo, ya que las redes neuronales se consideran una caja negra, éstas no entregan información de las características de las configuraciones prohibidas. Esto motivó a buscar entre ellas algo que pudiera diferenciarlas. Así, se construyó el siguiente índice I_K para alguna ventana K de 3×3 .

$$I_K = \sum_{\{i,j\} \text{adyacentes}} |i - j|.$$

En la Figura 2.9 se ilustra el índice I_K .

Figura 2.9: Valores del índice I_K para dos ventanas.

El promedio del índice de las configuraciones vistas corresponde a 12.9. Mientras al hacerlo en el conjunto de configuraciones no vistas se obtiene 13.9, lo que significa que las configuraciones vistas tienen valores más homogéneos en las celdas. Más aún, al calcular el índice I_K a A_4 , se tiene un promedio de 19.5, lo cual confirma el hecho que las configuraciones prohibidas son altamente heterogéneas. (Figura 2.10).

De manera similar, se propuso otro índice que busca diferenciar las configuraciones ob-

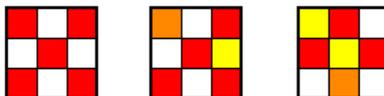
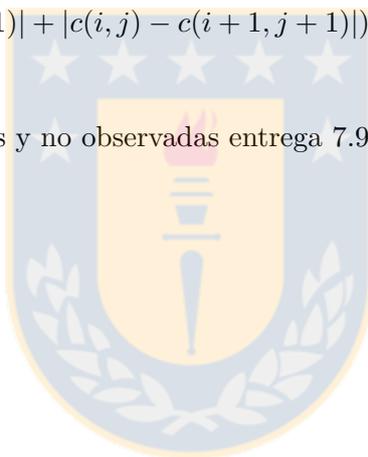


Figura 2.10: Configuraciones aisladas, con valores de I_K de 36, 31 y 27 respectivamente.

servadas. Este índice I_P suma la diferencia sólo con respecto a la celda central, ponderando por la distancia. Es decir

$$\begin{aligned}
 I_P := & |c(i, j) - c(i, j + 1)| + |c(i, j) - c(i, j - 1)| + |c(i, j) - c(i + 1, j)| + \\
 & |c(i, j) - c(i - 1, j)| + \frac{1}{\sqrt{2}} (|c(i, j) - c(i - 1, j - 1)| + |c(i, j) - c(i + 1, j - 1)| + \\
 & |c(i, j) - c(i - 1, j + 1)| + |c(i, j) - c(i + 1, j + 1)|)
 \end{aligned} \tag{2.1}$$

que para celdas observadas y no observadas entrega 7.97 y 8.56 respectivamente.



Capítulo 3

Modelamiento discreto de la dinámica delictual

3.1 Dinámica observada

Al graficar el número de crímenes para cada semana se observa que es bastante caótico (Figura 3.1 línea clara), aún dibujando una curva más suave correspondiente al promedio de una ventana de 7 semanas (línea oscura).

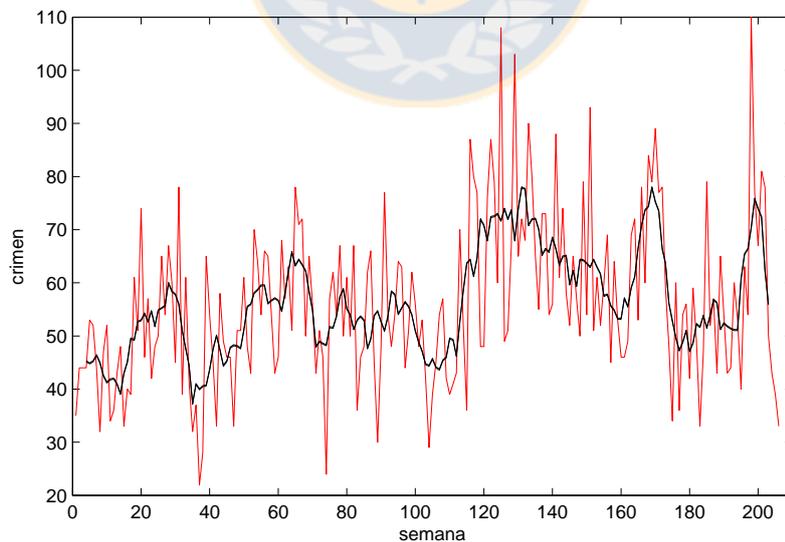


Figura 3.1: Crímen por semana en gris y el promedio de una ventana de 7 semanas en negro. Datos de Santiago.

Si observamos ahora la dinámica del crimen para cada celda en particular, el panorama se ve aún más caótico. Más todavía si consideramos que las celdas (2,2) y (2,1) son contiguas. En la Figura 3.2 se muestra la evolución de tres celdas en las primeras 40 semanas. Ya que se hace difícil de visualizar por la naturaleza discreta y el exceso de variación, en la Figura 3.3 se muestra el promedio de cada una de esas celdas pero promediándose en una ventana de 5 semanas.

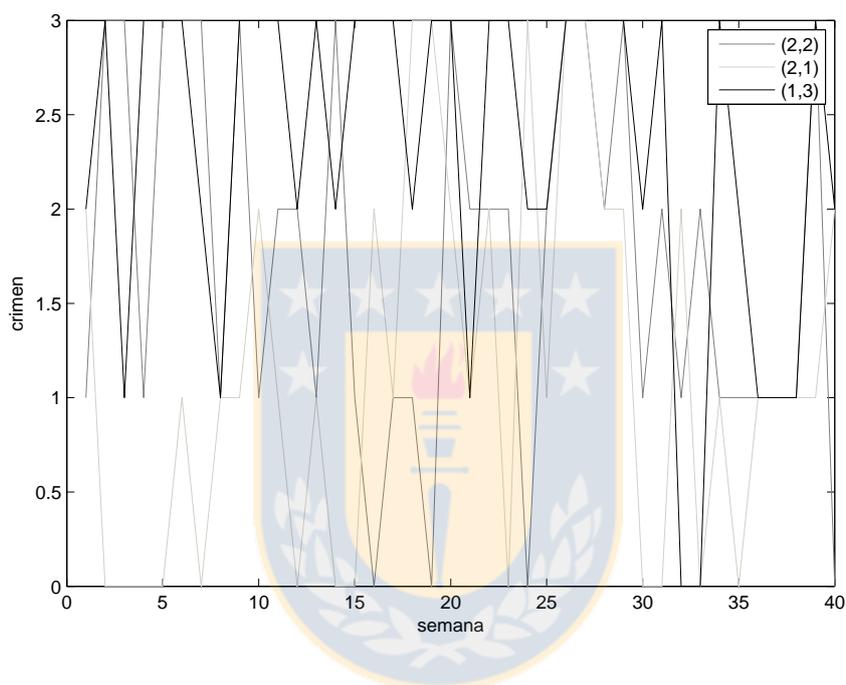


Figura 3.2: Crimen por semana para 3 celdas en particular durante las primeras 40 semanas. Datos de Santiago.

3.2 Autómata Celular

Para modelar la dinámica se decidió utilizar los Autómatas Celulares (AC), pues se ha mostrado que son capaces de modelar la dinámica delictual de buena manera [18]. Además, según las restricciones propias de los datos (cantidad y variedad) ya que el objetivo es encontrar un patrón común en el crimen, se decidió designar una misma función a todas las celdas que una a cada celda. Y como sabemos, esta es una característica propia de los AC.

Se decidió utilizar la vecindad de Von Neumann, es decir, la vecindad de $c_{i,j,t}$ es

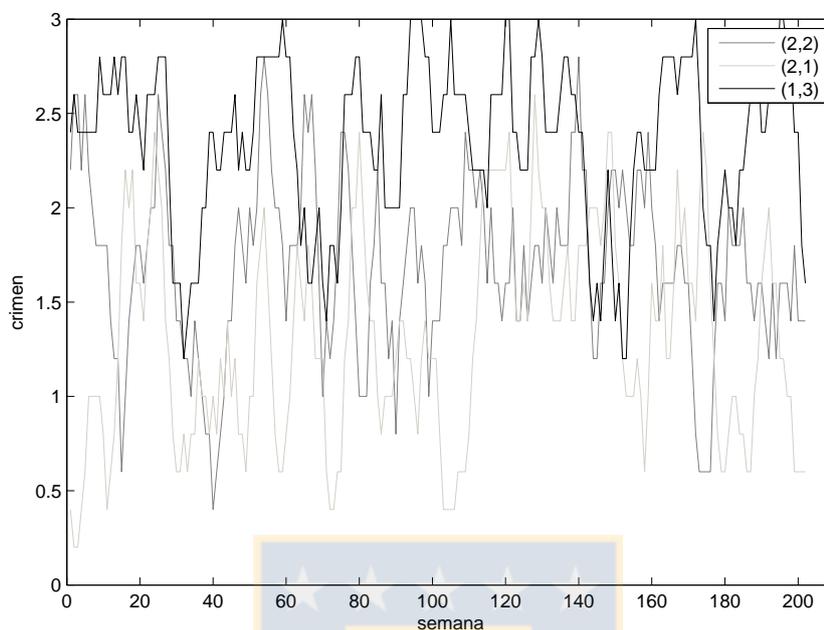


Figura 3.3: Crimen promedio en una ventana de 5 semanas para 3 celdas. Datos de Santiago.

$(c_{i+1,j,t}, c_{i-1,j,t}, c_{i,j+1,t}, c_{i,j-1,t}, c_{i,j,t})$, ya que otras vecindades arrojaron peores resultados. El contraste con otras vecindades se desarrollará en el capítulo siguiente.

En cuanto a las condiciones de frontera, se consideró una frontera abierta con valor 0. Ya que las fronteras geográficas de la zona que se analizó lo justificaba. Como es la presencia del río Mapocho en el norte, y la baja criminalidad en los otros extremos.

En cuanto a la función de transición del AC que modele el sistema, no es trivial elegirla, pues existen configuraciones que no se han observado y por otro lado, existen configuraciones que entregan distintas salidas en diferentes observaciones. Y dado que el autómata celular necesita una función de transición para todas las configuraciones de entrada, se debe encontrar una manera para, por un lado, decidir qué valor entregar a una condición inicial cuyas observaciones sean distintas, y por otro lado, conocer la salida de la función en configuraciones que no se hayan observado.

Para ello, se necesitó un método capaz de generalizar y completar. Se utilizaron las Redes Neuronales debido a su capacidad de generalizar funciones y su flexibilidad.

Es importante mencionar que de las 940 configuraciones distintas observadas, sólo el **22%** es función, es decir, siempre se observó la misma salida cuando se observó esa

configuración. Esto hace que sea imposible obtener un AC con esta vecindad (de von Neumann) con un error del 0%. En particular, el error mínimo alcanzable por un AC con vecindad de von Neumann es de **32%** para Santiago y **43%** para Ñuñoa.

3.3 Redes Neuronales Artificiales

La RN se entrenó con datos de una ventana móvil de vecindad de von Neumann por cada semana. Así, como la grilla es de 8×11 y se tienen 206 semanas, se alcanzan $88 \times 205 = 18040$ configuraciones. Considerando la vecindad de Von Neumann y los valores de c según se ilustra en la Figura 3.4, el conjunto de datos se construye de la siguiente manera:

$$(c_{i,j+1,t}, c_{i,j-1,t}, c_{i+1,j,t}, c_{i-1,j,t}, c_{i,j,t}) \rightarrow c_{i,j,t+1} \quad \forall i, j, t$$

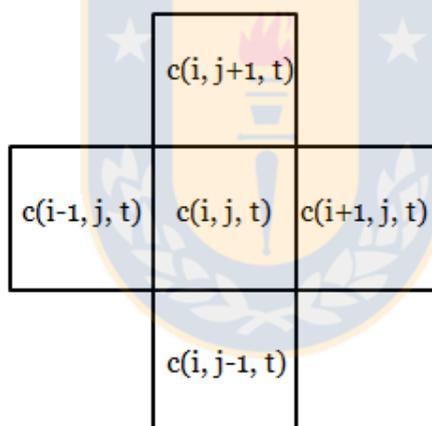


Figura 3.4: Valores de c para una semana t .

La distribución de las salidas no es homogénea. Como muestra la Figura 3.5, se tiene una alta concentración en celdas con valor 0, y mucho menos a medida que aumenta el crimen. Por la naturaleza de las RN, esto puede sesgar la salida. Por ello, no se consideraron los datos cuya entrada y salida fuese nula, logrando así una distribución más equilibrada. (Figura 3.6)

Además, tampoco se consideraron los datos de las celdas que mantuvieron su valor en 0, ya que no representa información útil.

Así, de estos datos, el 80% escogido aleatoriamente se utiliza para entrenar la red, y el

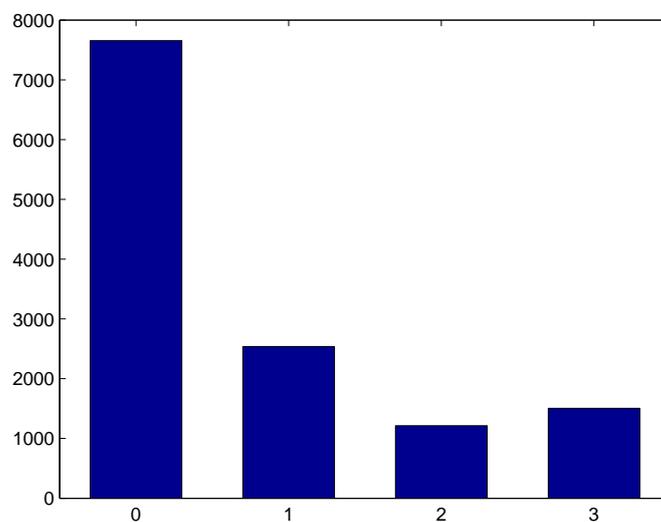


Figura 3.5: Histograma de valores de las celdas.

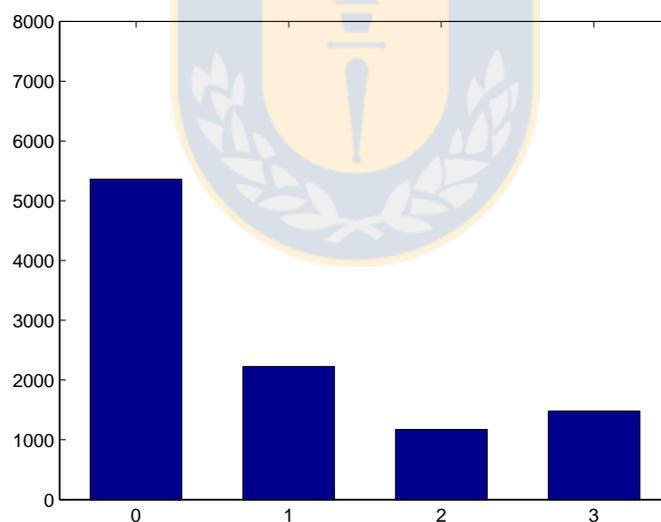


Figura 3.6: Histograma de las celdas excluyendo los casos cuyas entradas fuesen todas cero.

otro 20% como test. Esto con el fin de comparar los rendimientos en ambos conjuntos, y si el rendimiento en el conjunto de entrenamiento es mucho mejor que el de test, se puede concluir que la red en cuestión no pudo generalizar. En caso contrario, la red está más

bien aprendiendo de memoria las configuraciones observadas, lo cual no es de interés, pues como se comentó, el objetivo es conocer la dinámica delictual global, no sólo los casos observados.

Finalmente, contrastando distintas alternativas desechadas que se verán más adelante, la red neuronal se construyó con 5 neuronas de entrada, dos capas ocultas de 7 neuronas cada una y 4 neuronas de salida, donde cada una de ellas representa el valor 0, 1, 2 ó ≥ 3 (Ver Figura 3.7). A diferencia de utilizar una sola neurona de salida con valores entre 0 y 3, el hecho de separar las salidas hace que la red no busque promediar sus valores. Dado que la red busca minimizar el error cuadrático medio entre las salidas de la red con la salida esperada, en condiciones de mucho ruido, la red puede tender a entregar valores promedios.

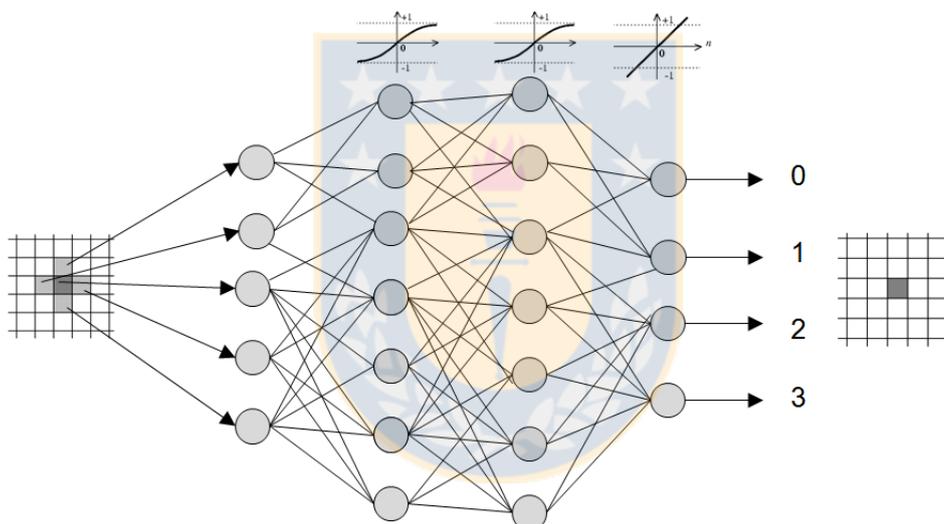


Figura 3.7: Arquitectura de la RN utilizada.

En la Tabla 3.1 se muestra el rendimiento de la red para distintas arquitecturas, considerando el factor no determinista de las RN. Junto al error de entrenamiento y de test, no se aprecian diferencias entre estos dos, lo que indica que la red neuronal no está sobrentrenada.

Notar que de estos AC, los marcados con asterisco (*) no entregaron salidas con valor 2 para los datos de test. Más aún, los otros AC entregan muy pocas salidas con valor 2. Esto muestra que las configuraciones que entregan salidas 2 no tienen un patrón reconocible por la red neuronal, por lo cual estas salidas son absorbidas por las demás.

Como ejemplo, en la Figura 3.8 A) se muestran las salidas esperadas del conjunto de

N_1	N_2	E. Train	E. Test		N_1	N_2	E. Train	E. Test	
3	3	0.5826	0.567	*	7	7	0.5834	0.5704	
3	3	0.5843	0.5684	*	7	7	0.5832	0.5714	*
3	3	0.5831	0.565	*	7	7	0.5825	0.5718	*
3	3	0.5812	0.5714	*	7	7	0.5853	0.5665	
3	3	0.5793	0.5704	*	7	7	0.5858	0.564	*
4	4	0.5799	0.5621	*	8	8	0.5838	0.5665	*
4	4	0.5842	0.5577	*	8	8	0.5816	0.567	*
4	4	0.5825	0.5694	*	8	8	0.5838	0.5704	*
4	4	0.5851	0.5714	*	8	8	0.582	0.5679	*
4	4	0.5815	0.567	*	8	8	0.5856	0.5577	*
5	5	0.5805	0.5684	*	9	9	0.5893	0.5513	
5	5	0.5809	0.5606	*	9	9	0.5845	0.563	
5	5	0.5871	0.5596	*	9	9	0.5856	0.5626	*
5	5	0.5842	0.5655	*	9	9	0.5849	0.5587	
5	5	0.5843	0.5684	*	9	9	0.5803	0.5635	
6	6	0.5838	0.5616	*	12	12	0.5826	0.5694	
6	6	0.5803	0.5626	*	12	12	0.5888	0.5767	
6	6	0.5816	0.5674	*	12	12	0.5894	0.5616	
6	6	0.5801	0.5699	*	12	12	0.588	0.566	
6	6	0.5851	0.5753	*	12	12	0.59	0.5591	

Tabla 3.1: Porcentaje de aciertos para distintas arquitecturas, un asterisco (*) indica que no se obtuvo valor 2 de salida en los datos de prueba.

entrenamiento, B) la salida del AC para este mismo conjunto y C) las salidas de todas las posibles configuraciones según el AC. Se observa como la red neuronal entrega mucho menos salidas 2 que lo observado. Lo contrario sucede con el valor 3.

Similarmente, para los datos de Ñuñoa (Figura 3.9) se tiene que la salida 2 y 3 tienen igual proporción en los datos observados, sin embargo, el AC construido con la red neuronal entrega una muy baja proporción de salidas iguales a 2.

Promediando los aciertos para cada arquitectura, se tiene el gráfico de la Figura 3.10. Así, a pesar que para 12 neuronas en cada capa oculta se tiene una mayor cantidad de aciertos en el conjunto de entrenamiento, no sucede lo mismo con el conjunto de test. Eso se debe a que al tener más neuronas, la red tiene mayor capacidad para memorizar las

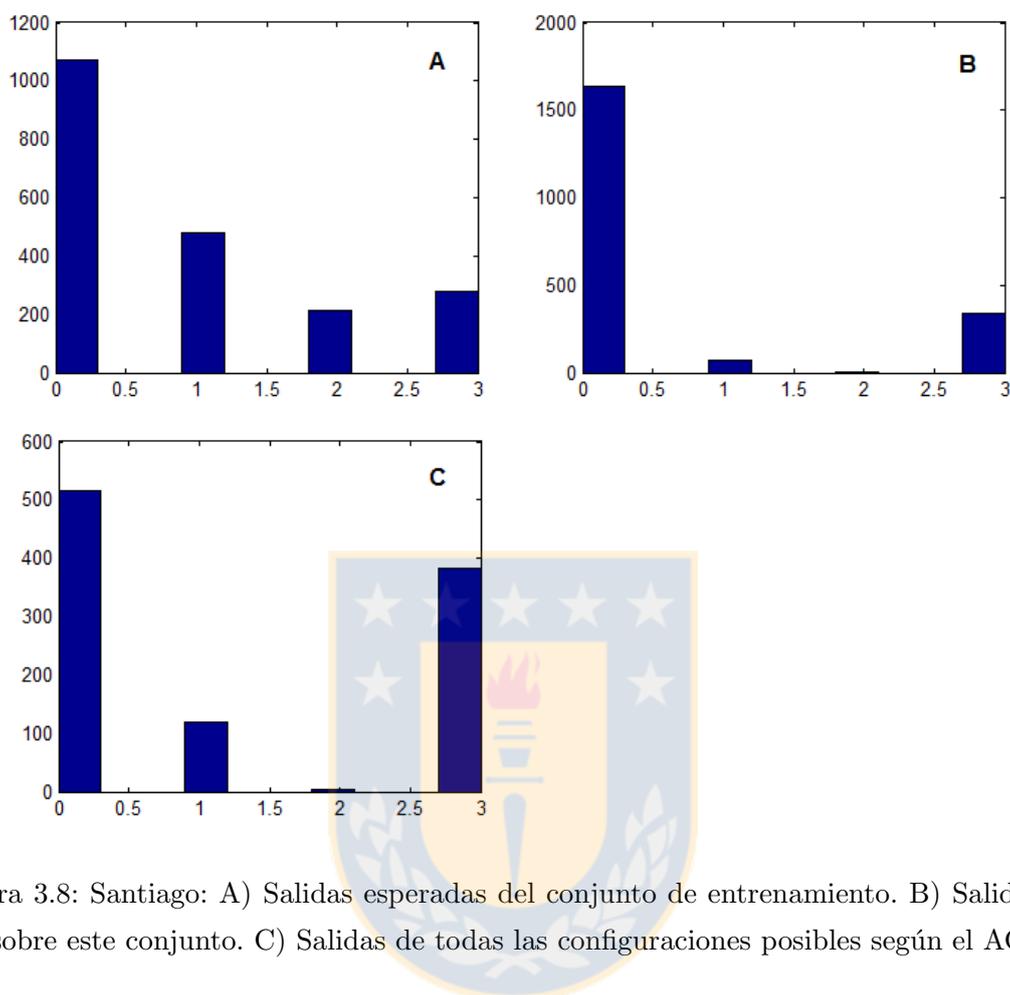


Figura 3.8: Santiago: A) Salidas esperadas del conjunto de entrenamiento. B) Salida del AC sobre este conjunto. C) Salidas de todas las configuraciones posibles según el AC.

entradas, pero su capacidad de generalizar se aprecia según los aciertos del conjunto de test. Sin embargo, la red con 7 neuronas en cada capa oculta obtuvo los mejores resultados en cuanto a error de test, superando por poco a las 3 neuronas por capa, además, al tener menos de 7 neuronas en cada capa, no se obtenían salidas 2. Por lo cual se eligió la arquitectura de 7 neuronas en cada capa oculta.

En el caso de Ñuñoa, se tiene un menor porcentaje de aciertos, como se observa en la Figura 3.11, y aunque la mejor arquitectura en cuanto a aciertos es la de 5 neuronas en cada capa, no se tienen salidas 2 en arquitecturas con menos de 7 neuronas. Por ello, se eligió, al igual que en Santiago, la arquitectura de 7 neuronas en cada capa oculta, pues es la con mayor aciertos dentro de las que entregan salida 2.

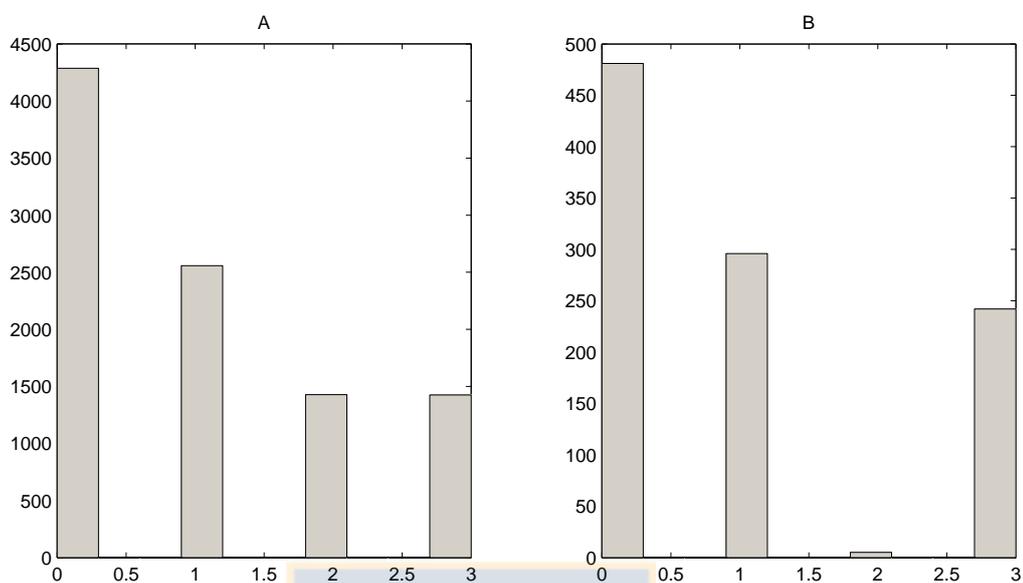


Figura 3.9: Ñuñoa: A) Salidas observadas B) salidas de todas las configuraciones posibles según el AC.

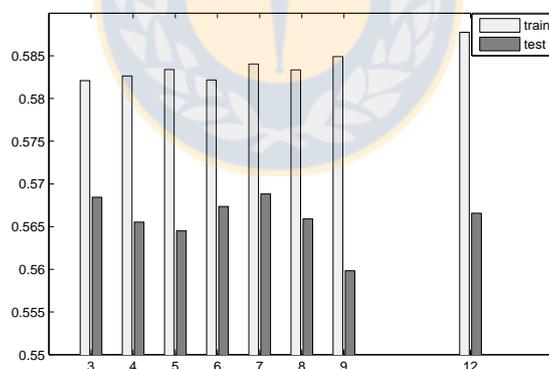


Figura 3.10: Santiago: Porcentaje de aciertos promedio para el conjunto de entrenamiento (*train*) y de test para cada red con dos capas ocultas, ambas con número de neuronas indicadas en el gráfico.

3.4 Desempeño del modelo

El error se calculó según el porcentaje de equivocaciones por celda por semana en promedio. Es decir,

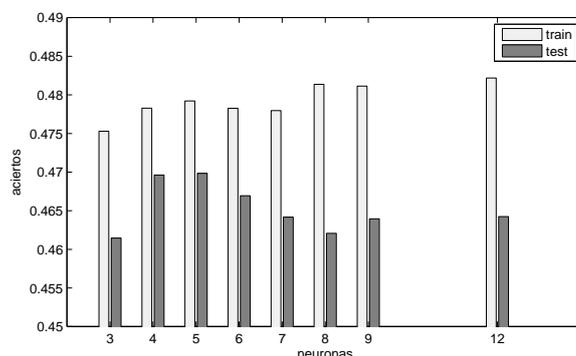
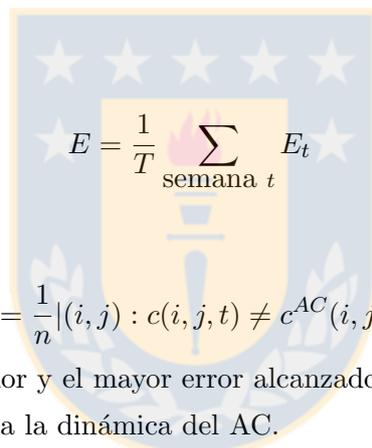


Figura 3.11: Para Ñuñoa: Porcentaje de aciertos promedio para el conjunto de entrenamiento (*train*) y de test para cada red con dos capas ocultas, ambas con número de neuronas indicadas en el gráfico.



donde

$$E_t = \frac{1}{n} |(i, j) : c(i, j, t) \neq c^{AC}(i, j, t)|$$

También se calculó el menor y el mayor error alcanzado en las semanas.

En la Figura 3.12 se ilustra la dinámica del AC.

Otra manera de contrastar la dinámica delictual con la entregada por el modelo es comparar ambas matrices de transición. Sin embargo, por un lado tenemos las transiciones observadas del total, es decir, por las 206 semanas y 63 celdas importantes y por el lado del modelo, conocemos todas las transiciones posibles a partir de las $4^5 = 1024$ configuraciones. Entonces, para que las matrices sean comparables, por el lado del modelo, sólo se consideraron las transiciones sobre las configuraciones observadas, mientras que del total de una misma configuración, se asignó como salida el valor que más veces haya sido salida de esta configuración. Así, en la Tabla 3.2 se muestran las matrices de transición de las configuraciones observadas y en la Tabla 3.3 el modelo.

3.5 Otros modelos

Además de los modelos basados en AC, podrían construirse muchos otros sencillos y naturales que podrían modelar la dinámica delictual.

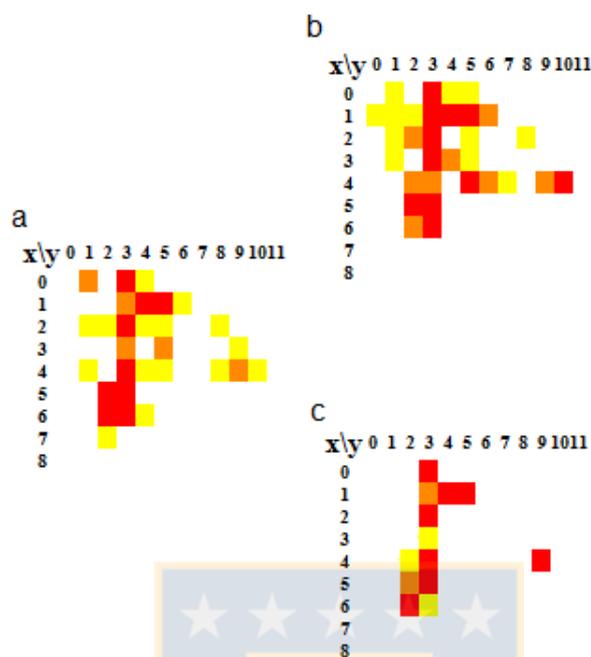


Figura 3.12: A partir de una semana a) se calcula la semana siguiente c) con el AC y se contrasta con la semana siguiente b). Modelo de Santiago.

Markov: Una primera aproximación sería, a partir del valor de la celda en una semana dada, predecir la siguiente de manera estocástica según lo que se haya observado. En la Tabla 3.4 se tiene la matriz de transición de estados. Por ejemplo, se observó 1249 veces que una celda con valor 1 cambiara a valor 0. Así, este modelo calculará la probabilidad de cambiar de estado i a j como

$$\frac{m(i, j)}{\sum_i m(i, j)}$$

donde m es la matriz de transición.

En la Tabla 3.5 se tiene la probabilidad de transición p , donde $p(i, j)$ indica la probabilidad de pasar del estado i al estado j .

En la Figura 3.13 se ilustra la dinámica del modelo markoviano.

Identidad: Otra manera sencilla para intentar modelar la dinámica delictual, es considerar que el crimen no cambia en una semana. Es decir, como el autómata *identidad*. Este sería el modelo nulo, ya que se espera que la dinámica encontrada sea mucho más compleja.

S	0	1	2	≥ 3	total	\tilde{N}	0	1	2	≥ 3	total
0	120	65	23	19	227	0	126	61	35	20	242
1	84	64	31	61	240	1	84	72	43	52	251
2	47	47	45	87	226	2	48	67	49	74	238
≥ 3	9	34	31	172	246	≥ 3	30	50	38	125	243
total	260	210	130	339	939	total	288	250	165	271	974

Tabla 3.2: Matrices de transición de las configuraciones observadas para (S) Santiago y (\tilde{N}) Ñuñoa, considerando la salida más común para una configuración que haya tenido salidas distintas.

S	0	1	2	≥ 3	total	\tilde{N}	0	1	2	≥ 3	total
0	191	27	6	3	227	0	183	58	1	0	242
1	129	62	10	39	240	1	106	117	0	28	251
2	61	63	16	86	226	2	73	99	0	66	238
≥ 3	9	32	16	189	246	≥ 3	41	42	0	160	243
total	390	184	48	317	939	total	403	316	1	254	974

Tabla 3.3: Matrices de transición del modelo de (S) Santiago y (\tilde{N}) Ñuñoa, considerando sólo las configuraciones vistas.

Promedio histórico: En el mismo sentido, está considerar al promedio histórico de cada celda, es decir, suponer que cada celda mantendrá en promedio su valor. Sin embargo, dado que lo que buscamos es maximizar el número de aciertos, al promediar los valores no se tiene el valor que minimice el error de una celda sin considerar sus vecinos. Para ello, debe considerarse el número de crímenes más común en ella, es decir, la **moda histórica**. Estos dos últimos modelos entregan un valor fijo independiente de la semana y no son AC pues dependen de la celda en particular.

AC elemental: Un modelo natural para construir la función de transición de un AC con la misma vecindad que utilizamos para el modelo (la de von Neumann) es entregar para cada configuración observada, el valor de salida que más se repita. Este es, de todos los AC para una vecindad fija, el que mayor número de aciertos puede alcanzar. Sin embargo, la falencia de este método es su baja capacidad para generalizar y además no es capaz de prevenir el valor de una configuración que no se haya observado.

	0	1	2	≥ 3
0	3537	974	286	176
1	1249	645	341	301
2	381	317	233	286
≥ 3	189	290	308	719

Tabla 3.4: Matriz de transición de estados de Santiago.

	0	1	2	≥ 3
0	0.70	0.20	0.06	0.04
1	0.50	0.25	0.13	0.12
2	0.31	0.26	0.19	0.24
≥ 3	0.13	0.19	0.20	0.48

Tabla 3.5: Probabilidad de transición de Santiago.

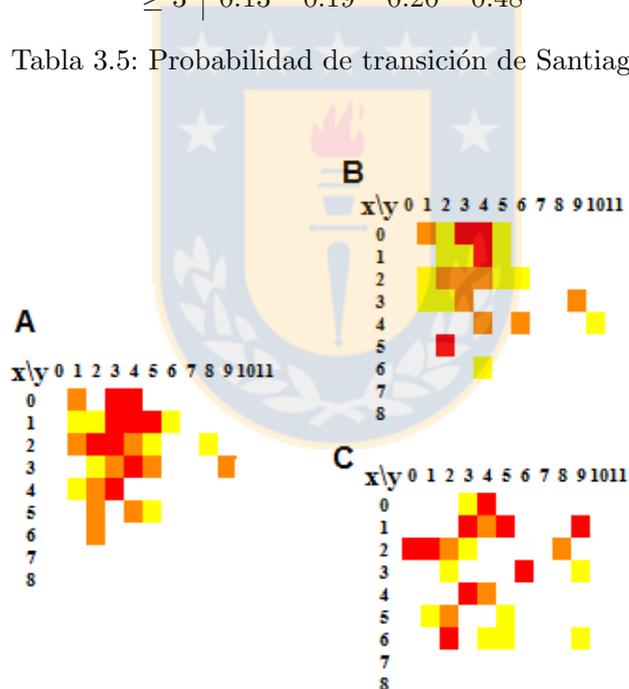


Figura 3.13: A partir de una semana A) se calcula la semana siguiente C) con el modelo markoviano y se contrasta con la semana siguiente B).

Moda modelo: La particularidad del modelo anterior con respecto al AC, es que éste es capaz de ver toda la historia, incluyendo el futuro, pues de esa forma calcula el valor fijo de cada celda. De manera más realista, se propuso un nuevo modelo que sólo revisara

las semanas anteriores de cada celda, cuya moda se utilizaría para predecir la semana siguiente. Para disminuir el número de variables a considerar, se consideró un modelo similar pero que sólo revisara las últimas 10 semanas (ó 5 ó 3).

Promedio vecinos: Una manera de lograr un modelo que pueda generalizar más, es prever el crimen según el promedio de las celdas vecinas (incluyendo la misma), de la semana inmediatamente anterior. Considerando la vecindad de Moore o la de Neumann.

Lineal De manera más general que promediar los vecinos, se pueden encontrar las mejores constantes α_{ii} que aproximen el valor de la semana siguiente a partir de las celdas vecinas (vecindad de von Neumann), por medio de minimizar el error cuadrático. Alternativamente, puede hacerse un modelo que sume una constante (**lineal + cte**).

AC Moore: Para contrastar el tamaño de la vecindad se construyó un AC que considerara la vecindad de Moore. Sin embargo, el resultado no mejoró pues la mejora que podría haber habido, no compensó el hecho de aumentar las dimensiones. En las tablas de denotará por *AC 9* debido a las celdas de su vecindad, al contrario de la vecindad de von Neumann que se notará por *AC 5*.

AC con historia Considerando el modelo presente, puede aumentarse la información histórica recibida, usando no sólo los crímenes de la semana anterior, si no también la anterior a ella. Esto duplicaría las dimensiones del problema, por lo tanto se consideró solo el valor central de la semana. Es decir, la vecindad de la celda (i, j) en la semana s quedará definida por

$$\{c(i-1, j, s-1), c(i, j-1, s-1), c(i+1, j, s-1), c(i, j+1, s-1), c(i, j, s-1), c(i, j, s-2)\}$$

La comparación de estos modelos se resumen en la Tabla 3.6.

De esta tabla, se observa cómo el mejor modelo en cuanto a porcentaje de aciertos es el AC elemental, que es lo esperado para cualquier modelo que considere la misma vecindad que él. Sin embargo, arroja más configuraciones prohibidas que el modelo propuesto. Además, este modelo no permite conocer la semana siguiente a partir de una semana que no se haya visto antes. En general, el AC elemental no logra generalizar ni encontrar algún

Método	Error	A_0
AC 5	0.36	2194
Markov	0.47	4700
AC 9	0.41	4600
AC historia	0.34	2494
Identidad	0.42	0
Prom. Hist.	0.39	4120
Moda hist.	0.33	2472
Prom. Vec5	0.48	4428
Prom. Vec9	0.48	4362
lineal	0.44	3934
lineal + cte	0.47	4362
AC elemental	0.32	2681
ModaModelo	0.34	1984
ModaMod10	0.35	2812
ModaMod5	0.36	3067
ModaMod3	0.39	2915

Tabla 3.6: Resumen del desempeño de distintos modelos alternativos para Santiago. En negrita el modelo propuesto.

patrón que se repita, si no sencillamente aprende de memoria lo observado y entonces no será capaz de adaptarse a una situación no vista.

La moda histórica también tiene buenos resultados en cuanto al error, pero no es comparable pues utiliza datos de todas las semanas para predecir. Sin embargo, la *moda modelo* entrega resultados ligeramente superiores al AC 5 y más aún, no presenta más configuraciones prohibidas. Aunque utiliza un gran número de variables para realizar el pronóstico.

Los modelos de la moda que utilizan menos variables, entregan valores similares, en cuanto a error, que el AC 5, pero el número de configuraciones prohibidas que entrega es mayor.

Como era de esperar, el modelo lineal da mejores resultados que el simple promedio, y ligeramente mejor si se suma una constante. Pero de todas formas, el resultado en cuanto a error y configuraciones prohibidas, está muy por debajo a los otros. Sin embargo, del modelo lineal puede rescatarse la información que entrega al calcular las constantes. Pues

da mayor importancia a la celda central que a las periféricas, como se observa en la Tabla 3.7.

Método	Error
$c(i-1, j)$	0.1784
$c(i, j-1)$	0.1027
$c(i, j)$	0.4537
$c(i, j+1)$	0.1123
$c(i+1, j)$	0.1092

Tabla 3.7: Constantes asignadas por le modelo lineal a cada celda de la vecindad. Para Santiago.

Método	Error	A_0
AC 5	0.502	2972
AC elemental	0.430	4094
ModaMod	0.480	3933
ModaMod10	0.496	4508
ModaMod5	0.517	5008

Tabla 3.8: Resumen del desempeño de distintos modelos alternativos para Ñuñoa. En negrita el modelo propuesto.

De los resultados de Ñuñoa (Tabla 3.8) se tiene, al igual que en Santiago, el modelo tiene menos aciertos que la *moda modelo*, pero ésta utiliza más datos que el AC. Las otras *moda modelo* con menos variables siguen siendo cercanas al modelo pero obteniendo más configuraciones prohibidas. La diferencia con Santiago es que en Ñuñoa la *moda modelo* tiene más configuraciones prohibidas.

Ahora, analizando con más detalle las configuraciones prohibidas según su pertenencia a los conjuntos A_i , podemos comparar los modelos de interés según el número de configuraciones aisladas como se observa en la Tabla 3.9. En ella se observa que el AC tiene menos configuraciones aisladas de A_0 , pero los otros modelos la superan en configuración más aisladas como en A_1 .

Modelo	$ A_4 $	$ A_3 $	$ A_2 $	$ A_1 $	$ A_0 $
ModaModelo	0	0	6	193	1984
ModaMod10	0	0	71	893	2812
ModaMod5	0	4	114	1034	3067
AC elemental	0	20	245	1143	2681
AC 5	0	12	173	874	2194

Tabla 3.9: Configuraciones aisladas para Santiago.

3.6 Dinámica y control en Santiago

A partir de la dinámica entregada por el modelo, se pueden encontrar distintos elementos, tales como puntos fijos y ciclos límites

Dinámica

Ciclos límite

Con el fin de comprender mejor el modelo creado, se buscaron elementos de interés, como puntos fijos de tamaño pequeño, ilustrados en la Figura 3.14 y otros ciclos límite de tamaño 2 (Figura 3.15), 3 (Figura 3.16) y 4 (Figura 3.17).

En general, se observa que los transientes son pequeño y no supera las 15 iteraciones. Mientras el período de los ciclos límites es menor a 5 en general.

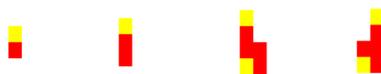


Figura 3.14: Algunos puntos fijos del AC.

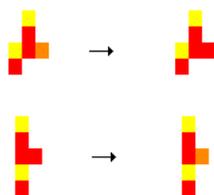


Figura 3.15: Configuraciones que tienen período 2.

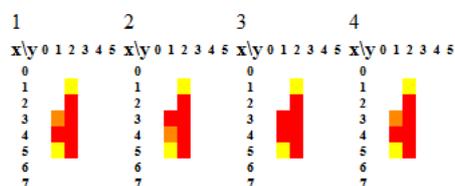


Figura 3.16: Configuración con ciclo límite de largo 3.

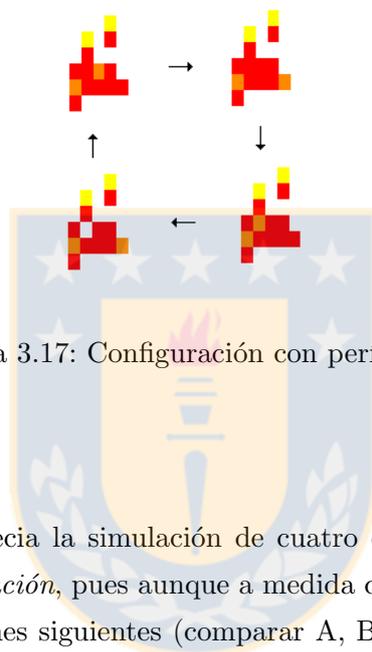


Figura 3.17: Configuración con período 4.

Saturación

En la Figura 3.18 se aprecia la simulación de cuatro configuraciones distintas, y se observa el fenómeno de *saturación*, pues aunque a medida que hay más crimen al inicio lo mismo sucede en las iteraciones siguientes (comparar A, B y C de la Figura 3.18), en el extremo de una gran concentración de crimen inicial (D) resulta en un menor número de crímenes final.

Intensidad

En la Figura 3.19 se muestra la influencia de la intensidad de inicial de una zona en el crimen total que se alcanza finalmente. Se observa cómo la influencia es directa y que a mientras mayor es la concentración inicial más crimen alcanza finalmente la zona.

Difusión

En la Figura 3.20 se contrasta la evolución de 3 configuraciones distintas, ordenadas de más nítida a más difuminada. Sin embargo, las 3 configuraciones tienen la misma cantidad de crimen inicialmente. Se observa cómo la difusión incide en una baja criminalidad a medida que pasa el tiempo, pues la configuración más difusa desaparece en sólo un paso

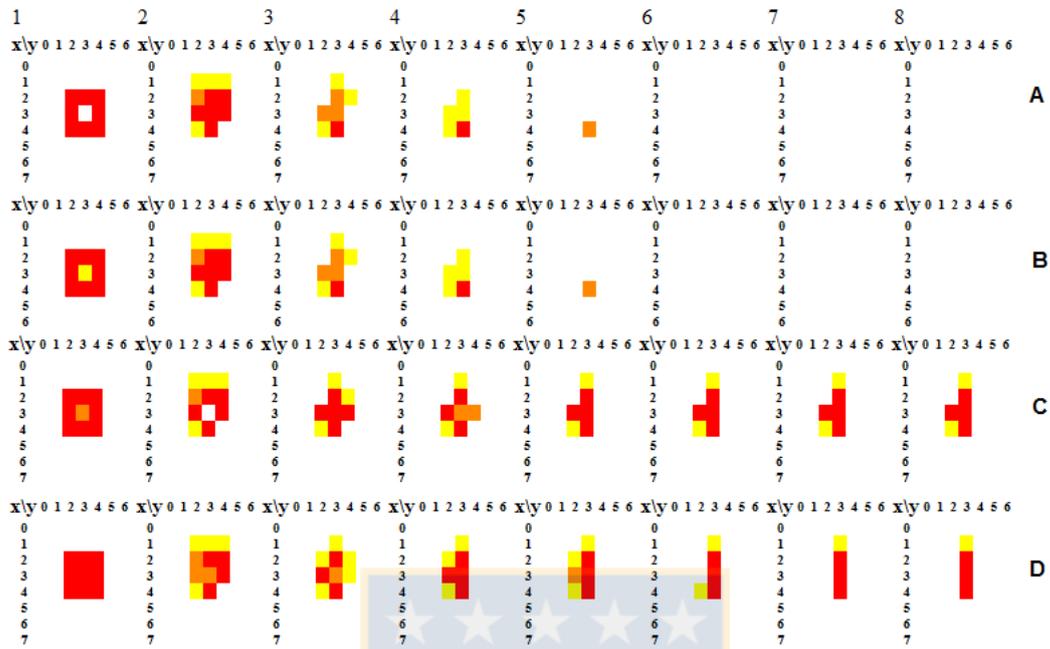


Figura 3.18: Simulación de cuatro configuraciones que difieren en sólo una celda.

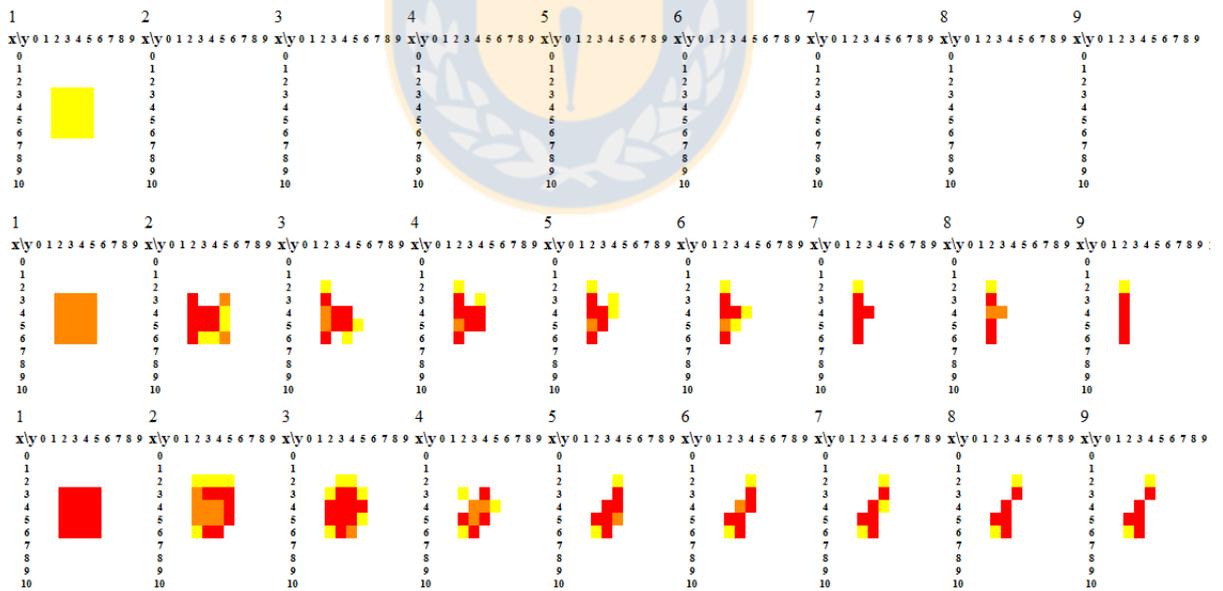


Figura 3.19: Distintas configuraciones que difieren en su intensidad de crimen.

de tiempo, mientras que, aunque la configuración más nítida finalmente desaparece, hasta

la décima iteración el crimen es mayor en ella en comparación con las otras dos.

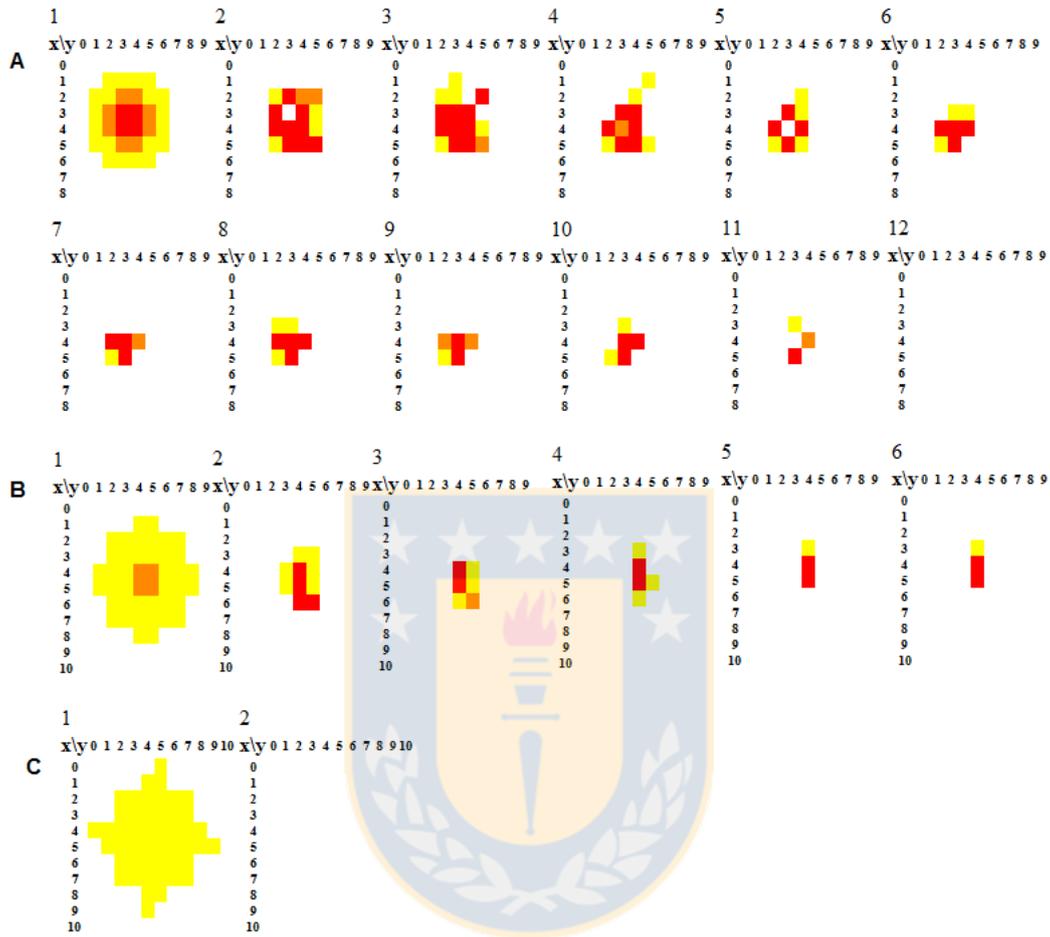


Figura 3.20: Distintos niveles de difusión y sus evoluciones en el tiempo.

Desmembrar

Al comparar las imágenes A y B de la Figuras 3.21 se observa que el AC no es simétrico. Además, si se desmiembra más (figura C) está configuración, el crimen no logra aumentar y formar un hotspot.

Control

Al aplicar el AC sucesivas veces a una configuración particular se obtiene lo que perdura en el tiempo si no hubiera influencias externas.

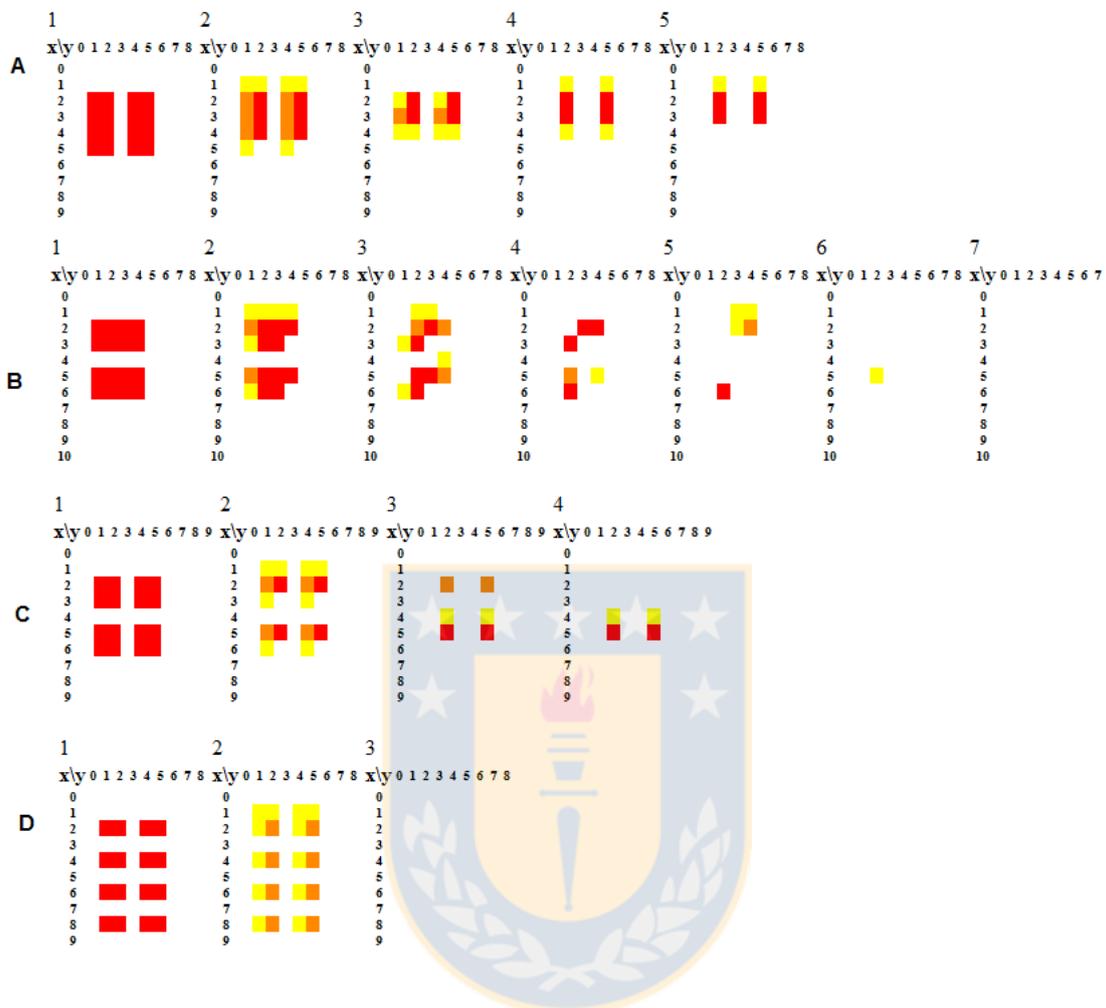


Figura 3.21: Desmembramientos de una configuración y sus evoluciones en el tiempo.

Para el caso de una semana real, se aplicó el AC repetidas veces a la semana 20, como se observa en la Figura 3.22. Si alteramos la configuración inicial de la semana cambiando una celda con valor 0 a 3, como se observa en un cuadro negro en la Figura 3.23, el estado estacionario presente una mayor cantidad de crimen y el transiente aumenta a su vez.

3.7 Dinámica y control en Ñuñoa

En el modelo de Ñuñoa existen menos puntos fijos pues las configuraciones tienden a desaparecer. Sin embargo, se encontraron algunos puntos fijos que se muestran en la Figura 3.24.

Si analizamos ahora el caso de Ñuñoa resultados similares en cuanto a que la intensidad

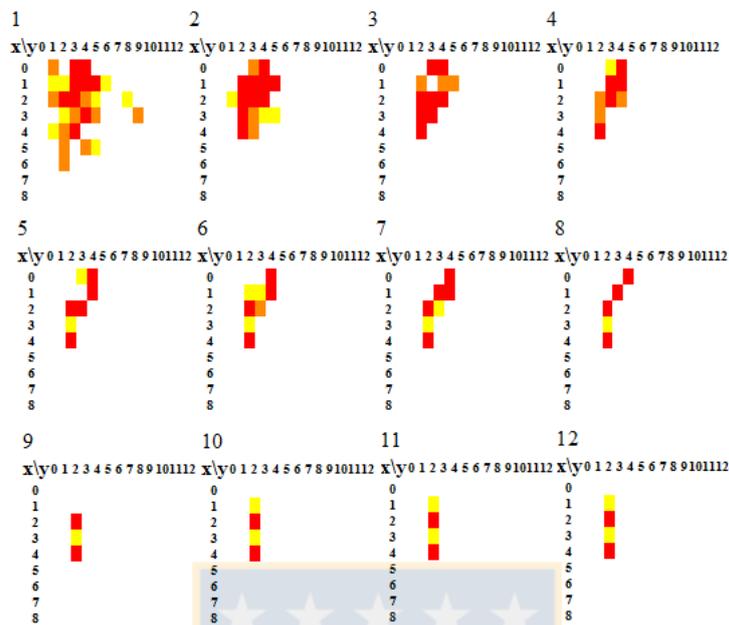


Figura 3.22: Simulación de la semana 20.

incide en un mayor crimen en las semanas siguientes (Figura 3.25) y un mismo número de crimen dispuesto de manera más difusa hace que éste desaparezca antes en el tiempo (Figura 3.26). Pero al contrario de lo que sucede en Santiago, el efecto de saturación se hace más evidente, pues al separar el desmembrar éste se mantiene fijo a través del tiempo (Figura 3.27 B y C). Además también se observa que no existe simetría al contrastar A con B de la misma figura.

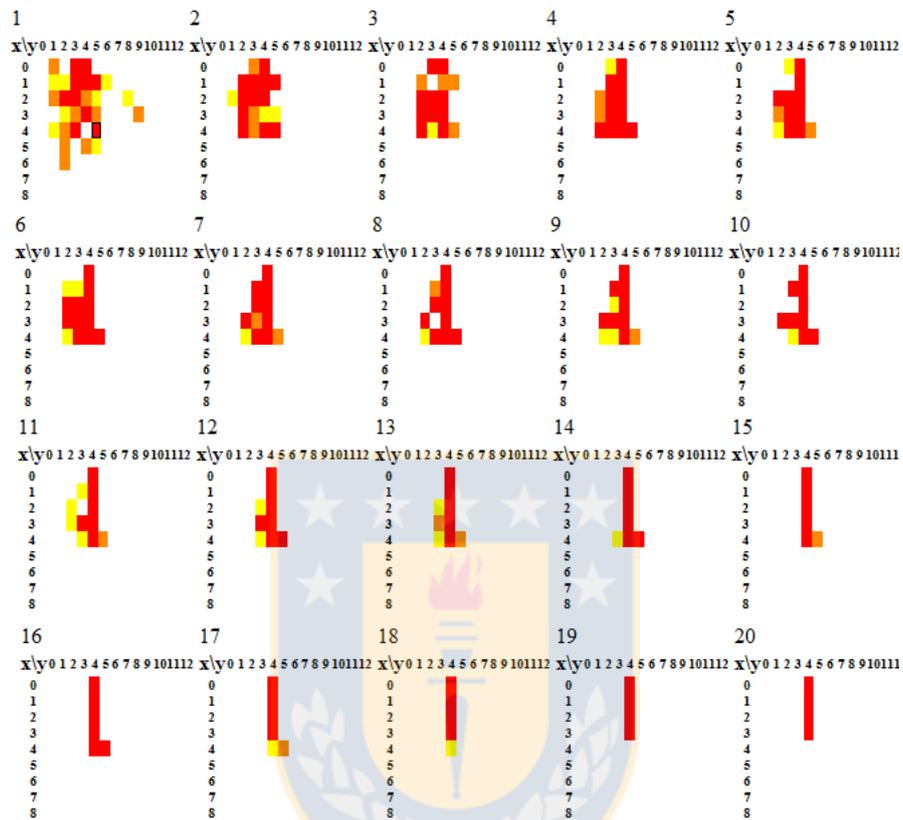


Figura 3.23: Simulación de la semana 20 alterando una celda, enmarcada en negro.



Figura 3.24: Algunos puntos fijos del AC de Ñuñoa.



Figura 3.25: Ñuñoa: Distintas configuraciones que difieren en su intensidad de crimen.

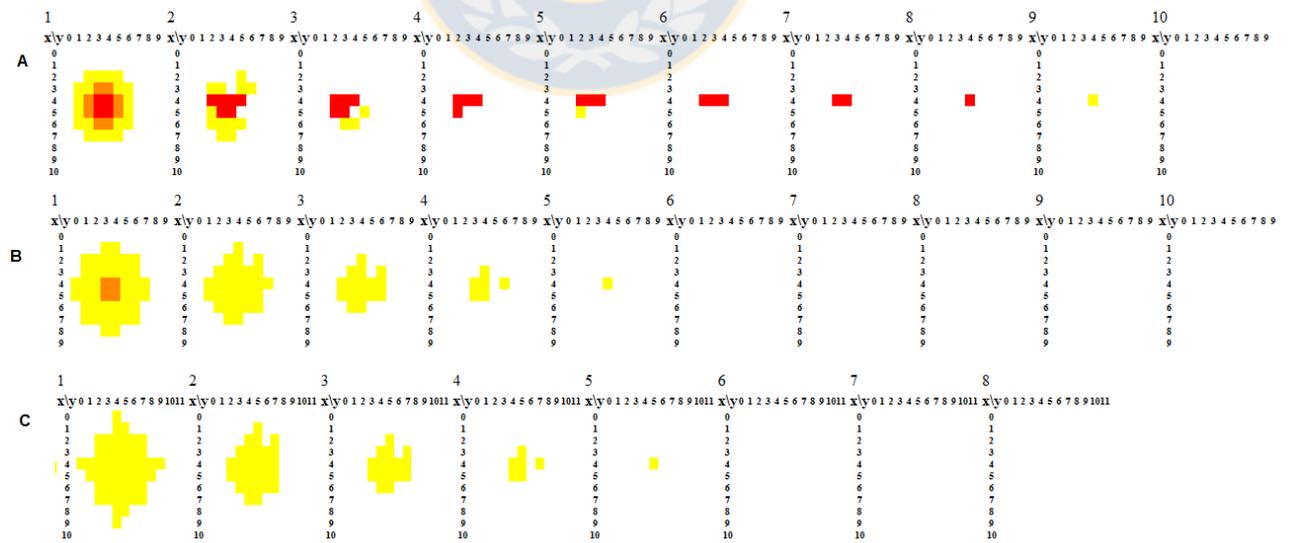


Figura 3.26: Ñuñoa: Distintos niveles de difusión y sus evoluciones en el tiempo.

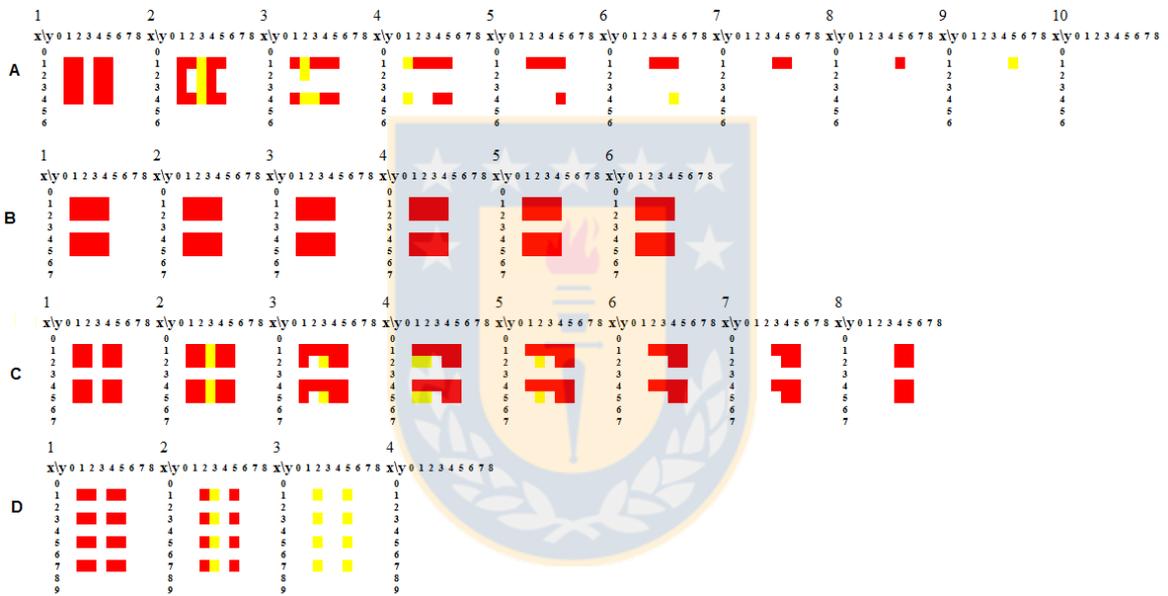


Figura 3.27: Ñuñoa: Desmembramientos de una configuración y sus evoluciones en el tiempo.

Capítulo 4

Herramientas computacionales

4.1 Matlab

Para trabajar con las redes neuronales, se utilizó Matlab, versión 7.8.0.347 (R2009a) junto al “Neural Network Toolbox” que facilita el uso de funciones para crear, entrenar y simular redes neuronales. En la Figura 4.1 se ilustra.

Por ejemplo, para el entrenamiento se ejecuta el siguiente código, donde $k1$ y $k2$ son el número de neuronas en las dos capas ocultas.

```
net=newff(train_in , train_out , [k1 k2] , { 'tansig ' 'tansig ' 'purelin ' });  
net=train(net , train_in , train_out );
```

```
test_out_red=sim(net , test_in );  
train_out_red=sim(net , train_in );
```

y teniendo la red, se construye el AC simulando la red con todas las configuraciones, como muestra la función “crear_automata”

```

function auto=crear_automata(net)
auto=zeros(1);
p=0;
k=3;
for i1=0:k
    for i2=0:k
        for i3=0:k
            for i4=0:k
                for i5=0:k
                    p=p+1;
                    s=sim(net,[i1 i2 i3 i4 i5]'); %simular
                    s=format(s); %cambiar formato
                    auto(p)=min(k,s); %guardar AC
                end
            end
        end
    end
end
end
end
end

```



4.2 PHP

La simulación, visualización y el procesamiento de datos se realizó con PHP 5.3.4. En el cuadro de abajo se muestra la función que obtiene la lista de configuraciones observadas a partir de los datos de la forma $\{c(i-1, j-1, t), c(i-1, j, t), \dots, c(i+1, j+1, t)\} \rightarrow c(i, j, t+1)$

```

function Vistos9() {
    $vistas=file("celdas_importantes9.txt"); //
    for($i=0;$vistas[$i];$i++) {
        $config=substr($vistas[$i],0,-4);
        $config_vistas[$config]=1;
    }
    return $config_vistas;
}

```

En el cuadro siguiente, la función que calcula, a partir de una semana (“\$celda”) y la función de transición del autómata celular (“\$auto”, la semana siguiente.

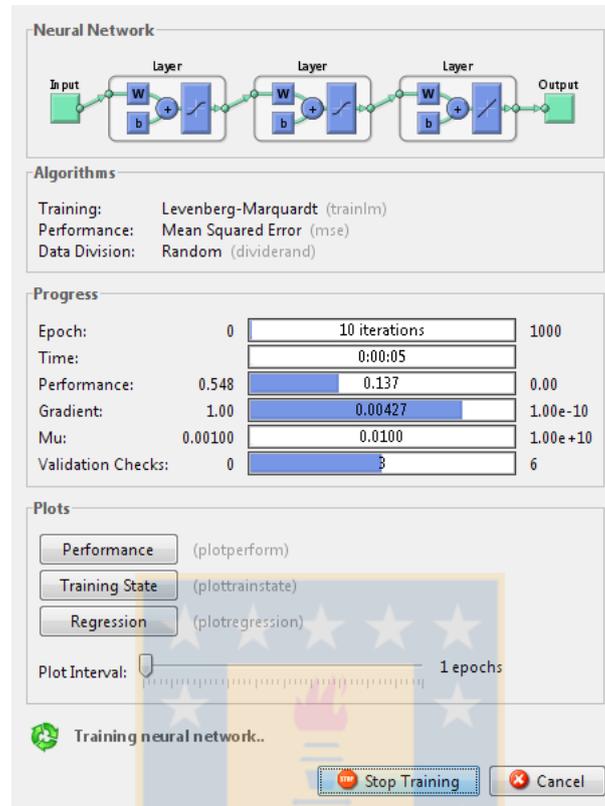


Figura 4.1: Ejemplo del Toolbox de RN de Matlab.

```
function simular($celdas,$auto,$nx,$ny) {
    for($i=0;$i<$nx;$i++) {
        for($j=0;$j<$ny;$j++) {
            $v8=$celdas[$i][$j-1];
            $v2=$celdas[$i][$j+1];
            $v4=$celdas[$i-1][$j];
            $v6=$celdas[$i+1][$j];
            $v5=$celdas[$i][$j];
            $celdas2[$i][$j]=$auto[$v8][$v4][$v2][$v6][$v5];
        }
    }
    return $celdas2;
}
```

Capítulo 5

Conclusiones

El modelo discreto construido obtuvo buenos resultados en cuanto a error y configuraciones prohibidas tanto en Santiago como en Ñuñoa, en comparación con los otros modelos. A pesar de no ser el mejor en error, sí resultó ser más balanceado en estos dos factores. La importancia de la vecindad está acotada, es decir, si no se considera, entrega un error más alto, mientras que si la vecindad es más grande, como ocurre con el modelo AC con vecindad de Moore, el error también es mayor. Utilizar constantemente el valor que más frecuentemente aparece en una celda es un modelo que entrega un error bajo, a pesar de su simplicidad. Sin embargo, se necesita conocer el futuro y en caso de adaptar el modelo para que revise sólo el pasado, no se alcanza un error significativamente menor que el modelo propuesto y la cantidad de configuraciones prohibidas es mayor.

El error mínimo alcanzado por un modelo determinista que considere la vecindad de von Neumann, es cercano al obtenido por el modelo propuesto: 32% y 36% respectivamente.

El modelo discreto construido permitió conocer características globales de la dinámica delictual. Se obtuvo que a medida que el crimen aumentaba entre una configuración y otra, éste aumenta hasta cierto nivel de concentración criminal, que hace que en el tiempo haya un fenómeno de *saturación*. La dispersión del crimen en una configuración; al igual que la fragmentación, aunque esta en menor medida; incide en una disminución del crimen en futuras iteraciones. Por otro lado, el hecho de aumentar en una ventana de tiempo hacia atrás el conocimiento de la celda disminuyó el error en el modelo de Santiago, no así en el modelo de Ñuñoa.

Para dos zonas distintas, tales como Santiago y Ñuñoa, se observaron dinámicas similares en algunos sentidos, como las características nombradas arriba. Pero en Ñuñoa no se observan fácilmente *hotspots* que perduren en el tiempo en las iteraciones del modelo, como sucede en Santiago. Además, en ambas regiones el valor 2 no presenta patrones

reconocibles por la red neuronal, por lo cual el AC rara vez entrega salidas de valor 2.

En cuanto a los patrones espaciales, se encontró que las configuraciones observadas no estaban distribuidas homogéneamente. En efecto, en el espacio de posibles configuraciones existen grandes zonas no visitadas. El número de configuraciones no visitadas en las simulaciones de los modelos, entregó información sobre la calidad de ellos para capturar patrones globales espaciales.

Trabajo futuro

En el desarrollo del modelo discreto se definieron algunos parámetros, como los estados, vecindad, topología de la red y de la grilla. Para lograr una eventual mejora en su rendimiento, estos parámetros podrían modificarse. Entre estos cambios se encuentran

- Cambiar la topología de la grilla, por ejemplo, a una grilla hexagonal.
- Utilizar otros estados, como $\{0, 1, 2, 3, \geq 4\}$ ó $\{0, \{1, 2\}, \geq 3\}$.
- Usar AC con memoria.
- Que el AC diferencie las celdas según parámetros que indiquen la naturaleza de cada celda.

El modelo propuesto podría construirse en otros casos para analizar las dinámicas de otros tipos de crímenes, tales como robo de autos, violaciones o tráfico de drogas. Similarmente, podrían analizarse otras zonas con distintas características. Así, en estas distintas situaciones se contrastaría el rendimiento del modelo.

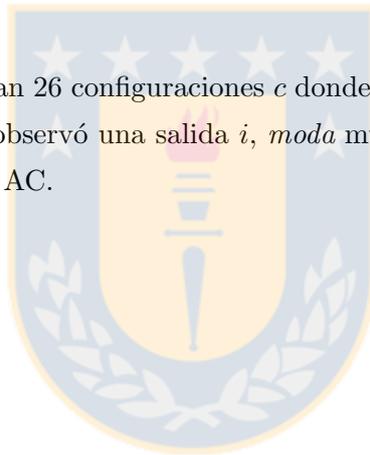
Un paso siguiente en la comprensión de la dinámica delictual, en particular de los *hotspots*, sería caracterizarlos a través del modelo discreto de acuerdo a las dinámicas globales observadas.

A lo largo del trabajo de esta Tesis, nos encontramos con diversos problemas combinatoriales ligados principalmente a la dinámica de los AC. La mayoría de ellos ya han sido estudiados y son conocidos ser NP-hard. Por ejemplo, la existencia de jardín del edén, biyectividad, alcanzabilidad de configuraciones, existencia de ciclos límite, etcétera [9]. Pero también algunos que, a nuestro conocimiento, no han sido tratados. En particular, el problema combinatorial de trayectorias (ver Apéndice B) podría estudiarse con mayor profundidad y así conocer si una matriz tiene solución. Además, podrían encontrarse condiciones para las matrices que faciliten saber si tiene solución y un algoritmo apropiado para encontrarla.

Apéndice A

Dinámica observada y del autómata para Santiago

En la Tabla A.1 se muestran 26 configuraciones c donde $s(i)$ indica el número de veces que a esa configuración se le observó una salida i , *moda* muestra la salida más común de ella y AC es lo que entrega el AC.



$c_{i+1,j}$	$c_{i,j-1}$	$c_{i,j}$	$c_{i,j+1}$	$c_{i-1,j}$	$s(0)$	$s(1)$	$s(2)$	$s(3)$	moda	AC
0	0	0	0	0	2300	313	47	23	0	0
0	0	0	0	1	510	65	15	7	0	0
0	0	0	0	2	152	32	12	4	0	0
0	0	0	0	3	100	22	8	7	0	0
0	0	0	1	0	371	79	13	2	0	0
0	0	0	1	1	74	21	2	0	0	0
0	0	0	1	2	26	12	4	1	0	0
0	0	0	1	3	9	5	1	2	0	0
0	0	0	2	0	147	33	4	3	0	0
0	0	0	2	1	38	4	1	1	0	0
0	0	0	2	2	9	2	0	1	0	0
0	0	0	2	3	4	1	1	1	0	1
0	0	0	3	0	142	27	3	2	0	0
0	0	0	3	1	17	10	1	1	0	0
0	0	0	3	2	5	2	3	1	0	0
0	0	0	3	3	0	2	0	0	1	0
0	0	1	0	0	290	72	31	19	0	0
0	0	1	0	1	67	27	10	4	0	0
0	0	1	0	2	31	12	6	2	0	0
0	0	1	0	3	24	14	5	3	0	0
0	0	1	1	0	57	24	9	7	0	0
0	0	1	1	1	25	7	2	2	0	0
0	0	1	1	2	7	3	2	0	0	0
0	0	1	1	3	8	6	4	3	0	0
0	0	1	2	0	40	11	4	3	0	0
0	0	1	2	1	8	5	2	1	0	0

Tabla A.1: 26 configuraciones observadas de Santiago.

Apéndice B

Problema combinatorial de trayectorias

Como vimos en el trabajo de tesis, a partir de la dinámica discreta observada, puede construirse una matriz de transición que contiene información de los estados de las celdas de la grilla. Ahora, nos preguntamos sobre la dificultad del problema inverso, es decir, dado una matriz de transición, queremos saber cuánta información sobre la dinámica podemos extraer, en particular, si es posible reconstruir una dinámica observada. Para ello, definimos el siguiente problema combinatorial.

Problema de Trayectoria: Dada una matriz M entera de $p \times p$ y un tamaño de grilla de $m \times n$ y un entero $T > 0$ tal que $\sum_{ij} M = (T-1)mn$ ¿Es posible construir una sucesión de T configuraciones de $m \times n$ cuya matriz de transición sea igual a M ?

Para ello, definamos primero el conjunto $K_p \forall p \in \mathbb{N}$. Si la matriz M cumple la propiedad de que la suma de la columna i sea igual a la suma de la fila i , es decir si

$$\sum_i M_{ij} = \sum_i M_{ji} \quad \forall j$$

diremos que $M \in K_0$. En general, si la matriz cumple

$$\sum_j \left| \sum_i M_{ij} - \sum_i M_{ji} \right| = 2p$$

entonces $M \in K_p$.

Observemos que si $M \in K_0 \cup K_1$, entonces interpretando cada estado como un nodo y a M como la matriz de pesos de los arcos, podemos construir un camino Hamiltoniano entre ellos y a partir de él construir los estados tales que su matriz de transición sea M .

El camino $a_1, a_2, \dots, a_{mn(T-1)+1}$ se puede cortar en mn partes de largo T cada uno para armar mn nuevos caminos, repitiendo el primer *nodo* en la última posición del camino anterior para mantener la matriz de transición. Así, se tiene

$$\begin{aligned} c^1 &:= \{a_1, a_2, \dots, a_T\} \\ c^2 &:= \{a_T, a_{T+1}, \dots, a_{2T-1}\} \\ &\vdots \\ c^{mn} &:= \{a_{mn(T-2)+1}, a_{mn(T-2)+2}, \dots, a_{mn(T-1)+1}\}. \end{aligned} \tag{B.1}$$

Luego, se debe construir el estado i como el i -ésimo elemento de cada camino. Como la matriz M es la de transición de estados, no interesa la posición de cada camino, pero los ubicaremos de derecha a izquierda y de arriba hacia abajo en la grilla.

Así la sucesión de configuraciones $\{S^t\}_{t \in \{1, T\}}$ resultaría

$$\begin{aligned} S_{1,1} &= c^1 \\ S_{1,2} &= c^2 \\ &\vdots \\ S_{1,n} &= c^n \\ S_{2,n} &= c^{n+1} \\ &\vdots \\ S_{m,n} &= c^{mn} \end{aligned} \tag{B.2}$$

y entonces S es una sucesión de estados cuya matriz de transición es M .

Por ejemplo, para $m = 2$, $n = 1$, $T = 4$ y una matriz

$$M = \begin{bmatrix} 1 & 2 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \end{bmatrix},$$

entonces $M \in K_1$ y se puede construir un camino $\{1, 1, 2, 3, 1, 2, 2\}$ Hamiltoniano. Luego se separa este camino en $c^1 = \{1, 1, 2, 3\}$ y $c^2 = \{3, 1, 2, 2\}$ y así, se tienen las siguientes 4 configuraciones de 2×1

$$\begin{bmatrix} 1 \\ 3 \end{bmatrix} \rightarrow \begin{bmatrix} 1 \\ 1 \end{bmatrix} \rightarrow \begin{bmatrix} 2 \\ 2 \end{bmatrix} \rightarrow \begin{bmatrix} 3 \\ 2 \end{bmatrix}.$$

Ahora, si $M \notin P_0 \cup P_1$ entonces si logramos construir mn caminos de largo T se podrán construir los estados de igual manera a cómo se hizo anteriormente. Equivalentemente, se está buscando si M puede separarse en a lo más mn matrices en $P_0 \cup P_1$ cuya suma de pesos sea múltiplo de $T - 1$.

Ahora, si $M \in P_k$ con $k > mn$ es fácil ver que es imposible separar M en menos de $mn + 1$ matrices en $P_0 \cup P_1$.

En conclusión, se sabe que si $M \in P_0 \cup P_1$ entonces existe una sucesión de estados cuya función de transición es M . Y por otro lado, si $M \in P_k$ con $k \geq mn$ entonces no existe tal conjunto de estados. Sin embargo, para $k \in \{2, \dots, mn - 1\}$ no se sabe cómo abordar el problema pues no se sabe si tiene solución. Esta pregunta queda abierta para ser respondida en el futuro.



Bibliografía

- [1] M. ANTHONY, *Discrete mathematics of neural networks: selected topics*. SIAM monographs on discrete mathematics and applications. 2001.
- [2] M. BARTHÉLEMY, J-P. NADAL Y H. BERESTYCKY, *Disentangling collective trends from local dynamics*. PNAS 107 (17) (2010), 7629-7634.
- [3] R. BERK, *Asymmetric Loss Functions for Forecasting in Criminal Justice Settings*. J Quant Criminol 27 (2011), 107-123
- [4] A. A. BRAGA, *The effects of hot spots policing on crime*. Ann Amer Acad Polit Soc Sci 578 (2001), 104-125.
- [5] P. J. BRANTINGHAM Y P. L. BRANTINGHAM, *Criminality of place: Crime generators and crime attractors*. Euro. J. Criminal Policy Res. 3 (1995), 5-26.
- [6] G. CYBENKO, *Approximations by superpositions of sigmoidal functions*. Mathematics of Control, Signals, and Systems, 2 (4) (1989), 303-314.
- [7] V. DABBAHIAN, P. JACKSON, V. SPICER Y K. WUSCHKE, *A cellular automata model on residential migration in response to neighborhood social dynamics*. Mathematical and Computer Modelling 52 (2010), 1752-1762.
- [8] D. DEADMAN, *Forecasting residential burglary*. International Journal of Forecasting 19 (2003), 567-578.
- [9] A. ILACHINSKI, *Cellular Automata* World Scientific (2001).
- [10] M. GARDENER, *Mathematical Games - The fantastic combinations of John Conway's new solitaire game "life"*. Scientific American. (1970) 120-123.
- [11] E. R. GROFF Y N. G. LA VIGNE, *Forecasting the future of predictive crime mapping*. Prev Stud 13 (2002), 29-57.

- [12] J. LIANG, *Simulating Crimes and Crime Patterns Using Cellular Automata and GIS*. Ph.D. Thesis, Department of Geography of the College of Arts and Sciences, Universidad de Cincinnati, EEUU, 2001.
- [13] M. A. DE MENESES, A-L. BARABASI, *Fluctuations in network dynamics*. Phys. rev Lett. (2004) 92:028701.
- [14] M. MINSKY Y S. PAPERT, *Perceptrons*. MIT Press, Cambridge. 1969.
- [15] G. F. RENGERT, A. R. PIQUERO Y P. R. JONES, *Distance decay reexamined*. Criminology 37 (1999), 427-445.
- [16] F. ROSENBLATT, *The Perceptron - a perceiving and recognizing automaton*. Report 85-460-1, Cornell Aeronautical Laboratory, 1957.
- [17] D. RUMELHART, J. L. MCCLELLAND Y THE PDP RESEARCH GROUP. *Parallel Distributed Processing: explorations in the microstructure of cognition*. MIT Press, Cambridge. 1986.
- [18] W. GORR Y A. OLLIGSCHLAEGER, *Spatio-temporal Forecasting of Crime: Application of Classical and Neural Network Methods*. In: H. John Heinz III School of Public Policy and Management, Carnegie Mellon University. Retrieved 2/15, 2004, from the World Wide Web: <http://www.heinz.cmu.edu/wpapers/retrievePDF?id=199723>.
- [19] JH. RATCLIFFE, *A temporal constraint theory to explain opportunity-based spatial offending patterns*. J Res Crime Delinq 43(3) (2006), 261-291.
- [20] L. W. SHERMAN, *Hotspots of crime and criminal careers of places*. Eck JE, Weisburd D (eds) Crime and place. Crime prevention studies, Criminal Justice Press, Monsey, NY, 4 (1995), 35-52.
- [21] L. W. SHERMAN, P. R. GARTIN Y M. E. BUERGER, *Hot spots of predatory crime: routine activities and the criminology of place.*, Criminology 27 (1989), 27-55.
- [22] M. B. SHORT, A. L. BERTOZZI Y P. J. BRANTINGHAM, *Nonlinear Patterns in Urban Crime: Hotspots, Bifurcations and Suppresion*. SIAM J. Applied Dynamical Systems 9 (2) (2010), 462-483.
- [23] M. B. SHORT, P. J. BRANTINGHAM, A. L. BERTOZZI Y G. E. TITA, *Dissipation and displacement of hotspots in reaction-diffusion models of crime*. Proc. Natl. Acad. Sci. (PNAS) 107(9) (2010), 3961-3965.

- [24] M.B. SHORT, M.B. D'ORSOGNA, V. B. PASOUR, G. E. TITA, P. J. BRANTINGHAM, A. L. BERTOZZI Y L.B. CHAYES, *A statistical model of criminal behavior*. Mathematical Models and Methods in Applied Sciences 18 (2008), 1249-1267.
- [25] J. Q. WILSON Y G. KELLING, *Broken windows: The police and neighborhood safety*. The Atlantic Monthly 127 (1982), 29-38.

