Clasificación de patrones mediante el uso de una red neuronal pulsante

Christian Hernández-Becerra, Manuel Mejía-Lavalle

Centro Nacional de Investigación y desarrollo Tecnológico,
Departamento de Ciencias Computacionales, Cuernavaca, Morelos,
México
{chrishb, mlavalle}@cenidet.edu.mx

Resumen. Se muestra cómo, mediante el uso de una sola capa de neuronas pulsantes, más aún con una sola neurona, es posible hacer la clasificación de patrones, ya sea de una función binaria como la función XOR o bien de una base de datos con decenas de características. Se ocupa el modelo de *Izhikevich* para modelar el comportamiento de las neuronas pulsantes utilizadas. Principalmente se pretende explotar el uso de una sola neurona para lograr realizar clasificación, analizando el posible alcance. Los resultados obtenidos son alentadores.

Palabras clave: Redes neuronales pulsantes, neurona de *Izhikevich*, clasificación, función XOR, base de datos.

Patterns Classification Using Spiking Neural Networks

Abstract. It is shown how, through the use of a single layer of spiking neurons, even more so with a single neuron, it is possible to make the classification of patterns, either a binary function as the XOR function or a database with tens of characteristics. *Izhikevich* model is used for modeling the behavior of the used spiking neurons. Mainly intends to exploit the use of a single neuron to perform classification, analyzing the possible scope. Obtained results are encouraging.

Keywords: Spiking neural networks, *Izhikevich* neuron, classification, XOR function, database.

1. Introducción

El proceso de clasificación y reconocimiento son un par de las características del ser humano que evidentemente han jugado un papel importante en su evolución. Resulta de interés para la Inteligencia Artificial poder recrear estas importantes habilidades del ser humano de forma minuciosa.

Teniendo en cuenta que estos procesos se llevan a cabo en el cerebro, no resulta descabellado ocupar herramientas como las Redes Neuronales Artificiales (RNAs) como principal herramienta, dado que los procesos cognitivos del ser humano (así como el de los mamíferos en general) suceden en el cerebro mediante la comunicación de las neuronas.

Uno de los modelos que surgen de las RNAs son los conformados por neuronas denominadas de tercera generación [1] cuya principal característica es la similitud que tienen con las neuronas que conforman el cerebro desde la perspectiva biológica. De forma más específica, las neuronas artificiales de este tipo simulan un voltaje de membrana y, al igual que las neuronas biológicas, arrojan una cantidad de pulsos a lo largo del tiempo. Es mediante estos pulsos que las neuronas se comunican y realizan determinados procesos.

1.1. Redes neuronales

Se pueden distinguir fácilmente tres posibles generaciones las RNAs. La primera o llamada Perceptron, basada en el modelo de McCulloch-Pitts[2], como unidades computacionales (0 y 1). El rasgo característico es que solo pueden dar como salida un dígito. Sin embargo cada función booleana se puede calcular por algunas multicapas de Perceptron con una sola capa oculta. La segunda generación se basa en el mismo modelo pero aplican una "función de activación"de imagen continua a una combinación de las entradas, la más común es la función sigmoidea[3]. Finalmente las de tercera generación son las llamadas pulsantes, que mediante una descripción matemática modelan con más realismo las neuronas biológicas [1].

1.2. Modelo *Izhikevich*

Existen varias estructuras de ecuaciones matemáticas que pretenden modelar el comportamiento de las neuronas artificiales pulsantes en respuesta al voltaje que reciben: Integrate-and-fire, FitzHugh-Nagumo, HindMarsh- Rose, Hoodgkin-Huxley, *Izhikevich*, entre otros [4].

En la referencia [4] se exponen dichos modelos, así como una comparación entre ellos para aportar un argumento sobre cuál de los modelos conviene utilizarse en términos de sus características. La Figura 1 ofrece una forma de comparación sencilla entre los modelos abordados en [4].

De esta manera es que se decide ocupar el modelo de *Izhikevich*, el cual se conforma de las Ecuaciones 1 y 2, y la condición de disparo de la Ecuación 3:

$$C\frac{dv}{dt} = k(v - v_r)(v - v_t) - u + I, \tag{1}$$

$$\frac{du}{dt} = a(b(v - v_r) - u), \tag{2}$$

$$\text{Si } v \geq v_{peak} \;\Rightarrow\; v = c,\; u = u + d. \tag{3}$$

De dichas ecuaciones se tienen las variables de la Tabla 1.

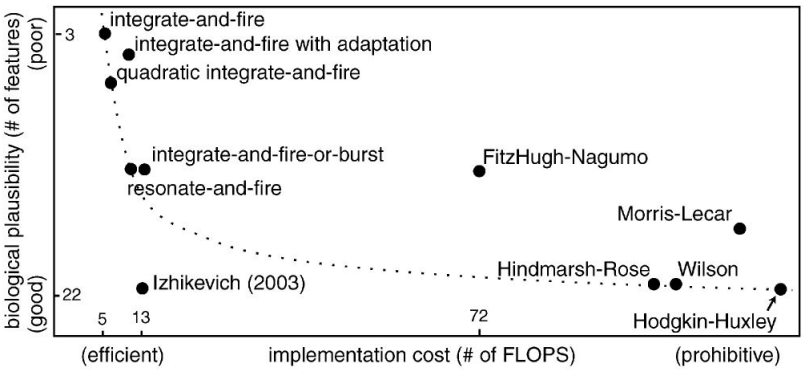

Fig. 1. Comparación entre la verosimilitud biológica y la eficiencia en términos de la fidelidad al comportamiento biológico y el costo de implementación [4]. Se observa que *Izhikevich* es la mejor opción

Como es conocido, una de las partes fundamentales de las RNAs es el entrenamiento de la red. Es importante mencionar que a diferencia de los modelos de primera y segunda generación, es ligeramente más difícil realizar el entrenamiento de la red por métodos tradicionales, lo que resulta es que en el modelo de neuronas de tercera generación, se ocupan otros métodos. Mas adelante se explica qué herramientas se usan para realizar el proceso de entrenamiento.

La estructura del resto del artículo es la siguiente: en la Sección 2 se muestran las herramientas que son ocupadas a lo largo de esta publicación. En la Sección 3 se muestra el desarrollo para realizar la clasificación de la función XOR con una sola neurona. En la Sección 4 se muestra el desarrollo para realizar la clasificación de base de datos *Ionosphere* con una sola capa de neurona. Finalmente en la Sección 5 se concluye y se discuten trabajos futuros.

2. Descripcción de las herramientas ocupadas

Parte de lo que se pretende con este trabajo es mostrar un poco del alcance que tienen las neuronas pulsantes dentro de la clasificación por sí solas, es decir, de cierta forma se pretende vislumbrar qué tanto se puede lograr con un mínimo de neuronas y tener una idea más clara del poder que tiene dicha herramienta.

2.1. Una capa

Aun cuando una RNA consta generalmente una capa de entrada, una o varias capas ocultas y una capa de salida [5], es posible lograr la clasificación con una sola capa cuando se trata de un problema de clasificación simple. Sin embargo cuando el problema aumenta de complejidad, una sola capa de neuronas de primera o segunda generación puede complicar la interpretación de la solución.

Tabla 1. Descripción de las Variables de las Ecuaciones 1, 2 y 3

Variable	Significado
I	Voltaje de entrada a la neurona
a	Constante del tiempo de recuperación
b	Constante de sensibilidad de la neurona
u	Variable de recuperación
C	Capacitancia
d	Variable de restablecimiento después del disparo
v_r	Valor de voltaje de la neurona en reposo
v_t	Voltaje del umbral instantáneo
c	Voltaje de reinicio
v	Potencial de la neurona
k	Parámetro para la forma del pico
v_{peak}	Valor del umbral de disparo

Aquí se ocupará una sola capa de neuronas pulsantes donde la entrada estará codificada por una función I definida en términos de las características de los patrones, y la salida se representa mediante la cantidad de disparos que suceden cuando el voltaje de la neurona supera el umbral v_{peak}, esta variable la llamaremos s.

Pese a que se podría escoger una cantidad arbitraria de neuronas para la capa, se realizan las pruebas con el mínimo posible, es decir, una neurona.

2.2. Modelo *Izhikevich*

Para el modelo de la neurona se ocupan los valores sugeridos en [7] y en [6] que se muestran en la Figura 2 para recrear disparos tipo *chattering*.

En la Tabla 2 se muestran los valores que se visualizan en la Figura 2 se ocupan para describir los picos mencionados.

Hay tres variables que aparecen en la Tabla 1 y que no aparecen en la Tabla 2, a saber: I, el voltaje de la neurona; v potencial de neurona y u, el voltaje de recuperación. Para poder ingresar información (voltaje) a la neurona, cada patrón debe ser convertido en un voltaje de entrada por medio de una función I que se describe en la siguiente subsección. El voltaje de recuperación es innecesario para los efectos del cómputo de los pulsos de respuesta y el voltaje v es precisamente donde se cuenta las veces que alcanza el valor v_{peak}.

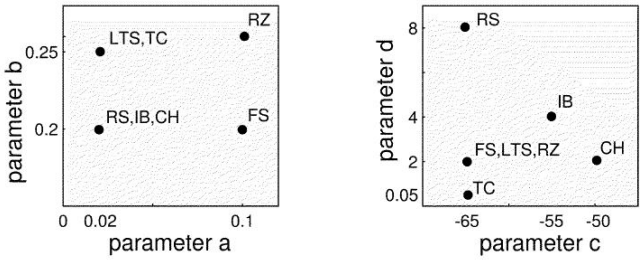

Fig. 2. Gráfica de los parámetros descritos en [7] y [6]. Los parámetros que interesan en este trabajo son lo que describen picos tipo *chattering*

Tabla 2. Valores de las variables del modelo de *Izhikevich*

Variable	Significado	Variable	Significado
a	= 0.03	v_t	= -40
b	= -2	c	= -50
C	= 100	k	= 0.7
d	= 100	v_{peak}	= 35
v_r	= -60		

2.3. Función de voltaje *I*

Algunas de las principales funciones que se ocupan para la codificación del voltaje se muestran en las Ecuaciones 4, 5 y 6. Sin embargo la que se utilizará en la experimentación es la función polinomial 4. En las tres ecuaciones, el valor de θ es el umbral mínimo que necesita la neurona para dar un disparo, pues no siempre es suficiente la codificación para que se logren dar pulsos [1].

Función polinomial. Es la función dada por la Ecuación:

$$I = (x \cdot w' + 1)^p + \theta. \tag{4}$$

Función productos. Es la función dada por la Ecuación:

$$I = (x \cdot w' \cdot \gamma) + \theta. \tag{5}$$

Función gaussiana. Es la función dada por la Ecuación:

$$I = e^{\frac{-||x-w'||^2}{2\sigma 2}} + \theta. \tag{6}$$

2.4. Algoritmo evolución diferencial

Este algoritmo cuya descripción detallada se encuentra en [9], es usado como una herramienta de aprendizaje no supervisado en [8]. De forma similar se ocupa aquí.

Como es conocido, para un algoritmo evolutivo se requiere una población, para este problema la población consiste de vectores conformados por pesos de la neurona. Es decir, cada elemento de la población del algoritmo evolutivo se interpreta como los pesos de red neuronal pulsante de una neurona. El algoritmo se describe en los párrafos siguientes.

El individuo i-ésimo de la generación G está denotado por $x_{i,G}$ y donde i es un valor entre 1 y NP, considerando que NP el numero de elementos en la población. En la primera generación los elementos de la población son creados de forma aleatoria (en nuestro caso siguiendo una distribución uniforme entre 0 y 1). Partiendo de la generación G, para la nueva generación se crean vectores de mutación $v_{i,G+1}$ se la siguiente manera: se eligen aleatoriamente 3 elementos de la población de la generación G denotados por $x_{r_1,G}, x_{r_2,G}$ y $x_{r_3,G}$ en donde queda claro que $r_1, r_2, r_3 \in \{1, ..., NP\}$. Se ocupa también una constante de *amplificación de diferencia de variación* $F \in [0, 2]$, finalmente se genera el vector de mutación con la ecuación 7:

$$v_{i,G+1} = x_{r_1,G} + F \cdot (x_{r_2,G} - x_{r_3,G}). \tag{7}$$

Note que para cada elemento de la población i se está generando un vector de mutación, eso es por que realiza un cruzamiento entre el el vector $x_{i,G}$ y el vector $v_{i,G+1}$ decidiendo con una probabilidad de $C_r \in (0, 1)$ si se hace o no el cambio sobre cada componente de los vectores. El vector resultante del cruzamiento se denota por $u_{i,G+1}$. Finalmente la selección de los elementos $x_{i,G+1}$ que conforman la siguiente generación está definida por la función de costo mínimo, es decir, si el vector $u_{i,G+1}$ produce el valor menor que el vector $x_{i,G+1}$ en la función de costo mínimo, entonces $x_{i,G+1} = u_{i,G+1}$ y su no, entonces $x_{i,G+1} = x_{i,G}$. Para nuestro problema, la función busca el máximo porcentaje de clasificación de los elementos representantes.

3. Ejemplo de clasificación, función XOR

Uno de los problemas más conocidos de clasificación es el de la función XOR (o también conocido como OR exclusivo) el cual no es posible clasificar mediante una función lineal, es decir, con el Perceptron. Aquí se ocupa una sola neurona con 2 parámetros de entrada para realizar la clasificación. Como es conocido, la función $XOR : \{0, 1\} \times \{0, 1\} \to \{0, 1\}$ se define como lo muestra la Tabla 3 , y su representación gráfica se puede apreciar en la Figura 3.

Una forma de resolver este problema de clasificación con neuronas de primera y segunda generación es utilizando 3 neuronas divididas en tres capas (dos en la capa oculta y una en la capa de salida). La red neuronal de una capa que se propone aquí para resolver este problema de clasificación consta de una

Tabla 3. Descripción de la función XOR

x_1	x_2	Clasificación
1	1	1
1	0	0
0	1	0
0	0	1

sola neurona, con 2 entradas. Se ocuparon como patrones de entrenamiento los elementos de la Tabla 3.

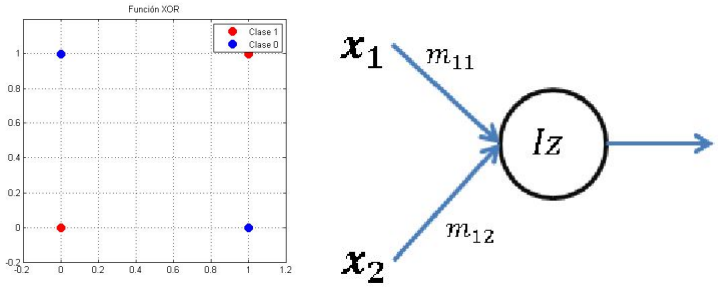

Fig. 3. Gráfica de la función XOR. Y la Neurona que se va a ocupar

Esta neurona se entrenó mediante el algoritmo de evolución diferencial. Dicho algoritmo, como ya se mencionó anteriormente, ocupa como función el porcentaje de elementos correctamente clasificados, es decir, que se realiza el entrenamiento buscando el 100 % de aciertos en la clasificación de lo patrones que se proporcionaron para entrenar la red (Los 4 elementos representados la Tabla 3).

La población para el algoritmo diferencial son vectores de dos dimensiones donde la componente i del vector corresponde al peso m_{1i}. Los valores m_{11} y m_{12} resultantes fueron en promedio (promedio de realizar 100 veces el mismo entrenamiento). En la Figura 4 se pueden ver los pulsos para los patrones del entrenamiento:

$$m_{11} = -0.9066 \quad \text{y} \quad m_{12} = -0.9089. \tag{8}$$

Finalmente con lo pesos arrojados por el algoritmo evolutivo mostrados en la ecuación 8 se tiene la red (conformada con una sola neurona) ya entrenada en donde la precisión de la clasificación es del 100 % de los elementos de entrenamiento.

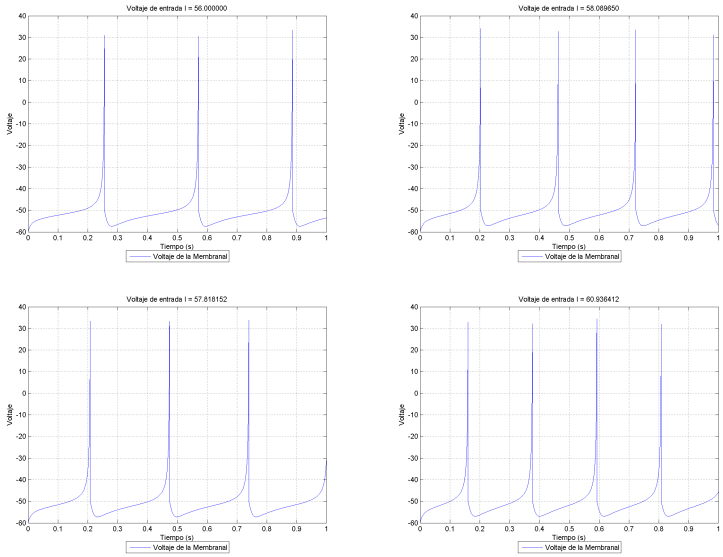

Fig. 4. Gráficas de pulsos de la neurona de *Izhikevich* para los elementos de entrenamiento

Para analizar la precisión de la clasificación, se puso a prueba la neurona con 400 patrones generados de forma aleatoria obtenidos aplicando una perturbación gaussiana sobre los patrones que se ocuparon para el entrenamiento, con una media de $\mu = 1$ ó 0 (si el valor es 1 ó 0 respectivamente), y varianza $\sigma = 0.1$. Se obtiene como resultado de clasificación un porcentaje del 97.5 % de patrones correctamente clasificados. La Figura 5 muestra la gráfica de la clasificación.

Con lo que se puede ver que la calidad de la clasificación es considerable dado que sólo se ocuparon 4 elementos para entrenar a la neurona. Evidentemente, si se proporcionan más elementos para el entrenamiento de la red neuronal, se logrará un porcentaje de clasificación más próximo al 100 %. En lo que sigue se realiza un análisis similar con una base de datos más compleja.

4. Datos de *Ionosphere*

4.1. Clasificación de *Ionosphere*

Se obtuvo la base de datos de *Ionosphere*.data que contiene información sobre electrones libres en la ionosfera. Los ecos de radar marcados como *buenos* son los que muestran evidencia de algún tipo de estructura en la ionosfera y los marcados como *malos* son aquellos que no. Estas señales tienen 17 pulsos y 2 atributos por cada impulso, por lo tanto cada instancia tiene 34 atributos. La base de datos contiene un número total de 351 instancias. En el atributo 35 se asigna la clasificación que puede ser *bueno* o *malo*. Con la descripción de la base

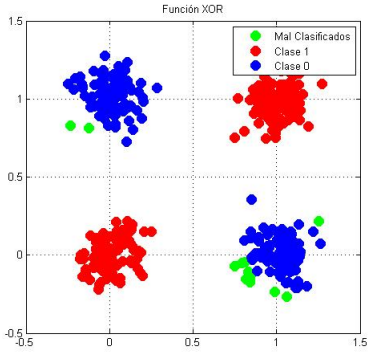

Fig. 5. Gráfica de factibilidad de clasificación de la neurona de *Izhikevich*

de datos se tiene que se ocupará una sola neurona con 34 parámetros de entrada para realizar la clasificación. Ver figura 6.

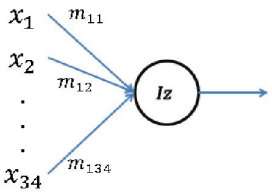

Fig. 6. Modelo gráfico de la neurona utilizada para la clasificación de los datos *Ionosphere*

Esta neurona se entrenó mediante el algoritmo de evolución diferencial de forma similar al problema de clasificación XOR, salvo por un paso intermedio. Como se puede observar, aquí no se cuenta con representantes, por lo que se seleccionó un porcentaje de la base de datos de forma aleatoria para realizar el entrenamiento. El porcentaje utilizado para el entrenamiento fue del 20 %. El algoritmo realizó el entrenamiento buscando el 100 % de aciertos de la población de muestra, es decir, de 70 elementos de la población total para el entrenamiento. Los parámetros del algoritmo evolutivo se describen en la Tabla 4.

Es importante notar que los parámetros del algoritmo diferencial se escogieron austeros para mostrar la simplicidad de condiciones necesarias. Los valores de los pesos que se encontraron con el algoritmo de entrenamiento permitieron una clasificación en promedio del 87.3 % en el conjunto de entrenamiento. En la Tabla 5 se detalla la cantidad promedio de disparos que se presentaron en las dos clases.

Tabla 4. Parámetros del algoritmo diferencial

Parámetro	Valor
Parámetro	Valor
Generaciones	$G = 100$
Total de la población	$NP = 40$
Tasa de diferencia de Variación	$F = 0.9$
Factor de recombinación	$C_r = 0.8$

Tabla 5. Resultados de la clasificación de *Ionosphere*

Clase	Medias de disparos (Representante de clasificación)
Malos	14.9682
Buenos	37.02222

Cuando se puso a prueba la red entrenada con los pesos encontrados mediante el algoritmo diferencial (que ya clasificaba correctamente el 87.3 % de los 70 elementos escogidos al azar para entrenar) se logró la clasificación correcta de 273 elementos de los 351, equivalente al 77.78 % de efectividad. Es decir que, en este caso, el 20 % de la población es suficiente para clasificar correctamente al 77.78 % de la población total.

5. Conclusiones y trabajo futuro

El desarrollo de una red multicapas es útil, pero en ocasiones hay problemas que pueden resolverse con menos "esfuerzo". Las neuronas pulsantes pueden ser ocupadas de forma muy simple en cuanto a (la estructura) obteniendo un resultado aceptable en la clasificación de patrones. Como se mostró en el presente trabajo, teniéndose resultados bastante alentadores. A lo largo del desarrollo de este trabajo se encontraron varias líneas que resultan interesantes y de las cuales se podrían obtener experimentos y observaciones con prometedoras conclusiones, por ejemplo en el problema de clasificación de la función XOR, resulta interesante indagar la forma de la separación del espacio de patrones cuando en el entrenamiento se presentan sólo los 4 representantes de la Tabla 3.

Es interesante analizar la clasificación lograda con la base de datos *Ionosphere*, y esto se podría hacer de forma similar en otras bases de datos. También

parece de interés averiguar qué tanto deben cambiarse los parámetros del algoritmo diferencial para lograr un mejor porcentaje de clasificación, o bien averiguar si hay algunos factores ajenos a la red neuronal que dificultan la clasificación (como diferentes subclases dentro de una misma clasificación).

Aparentemente, las neuronas pulsantes no ponen restricción a la clasificación binara (bueno o malo), es decir, que una línea de interés a futuro puede ser averiguar hasta dónde se puede clasificar con una sola neurona cuando se necesitan más de dos clases. Queda claro que el tema es muy basto y que aún cuando se descubren respuestas, siempre se encuentran nuevas preguntas.

Referencias

1. W. Maass: Networks of Spiking Neurons: The Third Generation of Neural Network Models. Institute for Theoretical Computer Science. Technische Universitit Graz, Neural Networks, Vol. 10, No. 9, pp. 1659–1671 (1997)
2. W. McCulloch, W. Pitts: A logical calculus of the ideas immanent in nervous activity. Bulletin of Mathematical Biophysics, 5:115–133 (1943)
3. D. E. Rumelhart, J. L. McClelland: Parallel distributed processing. Vol. 1, IEEE (1988)
4. E.M. Izhikevich: Which Model to Use for Cortical Spiking Neurons? IEEE Transactions On Neural Networks, Vol. 15, pp. 1063–1069 (2004)
5. W. Gerstner, W. Kistler: Spiking Neuron Models: Single Neurons, Populations, Plasticity. Cambridge University Press (2002)
6. E.M. Izhikevich: Simple Model of Spiking Neurons. IEEE Transactions on Neural Networks, Vol. 14, pp. 1569–1572 (2003)
7. I. C. Matadamas: Aplicación de las redes neuronales pulsantes en el reconocimiento de patrones y análisis de imágenes. Tesis de maestría. Centro de Investigación en Computación, Instituto Politécnico Nacional, México D.F. (2014)
8. R. Storn, K. Price: Differential Evolution-A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. Journal of Global Optimization, Vol. 11, pp. 341–359 (1997)
9. J. I. Espinosa-Ramos, N. Cruz-Cortes, R. A. Vázquez: Creation of Spiking Neuron Models Applied in Pattern Recognition Problems. In: Proceedings of International Joint Conference on Neural Networks, Dallas, Texas, USA (2013)
10. A. L. Hodgkin, A. F. Huxley: A quantitative description of membrane current and application to conduction and excitation in nerve. J. Physiol., vol. 117, pp. 500—544 (1954)
11. E. Izhikevich, G. M. Edelman: Large-Scale Model of Mammalian Thalamocortical Systems. PNAS, 105(9), pp. 3596–3598 (2008)