Revista Cubana de Ciencias Informáticas
ISSN: 1994-1536
rcci@uci.cu
Universidad de las Ciencias Informáticas
Cuba

Bonet Cruz, Isis; Salazar Martínez, Sain; Rodríguez Abed, Abdel; Grau Ábalo, Ricardo; García Lorenzo, Maria Matilde

Redes neuronales recurrentes para el análisis de secuencias

Revista Cubana de Ciencias Informáticas, vol. 1, núm. 4, 2007, pp. 48-57
Universidad de las Ciencias Informáticas
Ciudad de la Habana, Cuba

Redes neuronales recurrentes para el análisis de secuencias

Recurrent neural network for sequences analysis

Isis Bonet Cruz*, Sain Salazar Martínez, Abdel Rodríguez Abed, Ricardo Grau Ábalo y María M. García Lorenzo

Centro de Estudios de Informática, Universidad Central "Marta Abreu" de Las Villas, Carretera a Camajuaní, km 5 ½, Santa Clara, Villa Clara, Cuba. CP 54830

*Autor para la correspondencia: isisb@uclv.edu.cu

REVISTA CUBANA DE CIENCIAS INFORMÁTICAS VOL.1 No.4 DICIEMBRE 2007 p. 48-57

Resumen

En este trabajo se realiza un estudio de la importancia de las redes recurrentes y de su naturaleza en análisis de secuencias o señales donde es muy importante tener en cuenta el pasado o el futuro.

Se muestra la fortaleza de estos métodos para analizar secuencias de tamaño variable, por su despliegue en el tiempo en función del tamaño de la entrada. Se construyen específicamente redes neuronales recurrentes bidireccionales, como una especificación de redes recurrentes, mostrando la potencialidad de las mismas en sistemas no causales, donde las entradas pueden depender de entradas de tiempos pasados y futuros. Además se desarrolla una plataforma para implementar redes recurrentes dinámicas, con algoritmo de aprendizaje *Backpropagation Trough Time*; que permiten desarrollar redes para cualquier problema donde las entradas son secuencias analizadas en el tiempo y la salida son otras secuencias o simplemente descriptores de funciones o propiedades de las mismas.

Palabras clave: Clasificación, red neuronal recurrente, secuencias, sistemas dinámicos.

Abstract

In this paper, the recurrent neural networks importance is analyzed, as well as, its nature in sequence or signal analysis in problems, where the past or future features have to be taken into account. It's also shown the strength of these methods to analyze sequences with variable length, due to their unfolding in time. More specific, bidirectional recurrent neural networks were built taking advance of their effectiveness to solve causal systems where there are dependencies among input features through time. A platform is also developed to implement dynamic recurrent neural networks, with Backpropagation Trough Time as training algorithm. This software allows us to design and train networks with any topology, in problems where the inputs are sequences in time and the output can be other sequences, function descriptors or properties of the sequences.

Key words: *Classification, dynamic system, recurrent neural network, sequences.*

Introducción

El análisis de señales en el tiempo es estudiado en varias áreas como la Medicina, la Bioinformática, la Electrónica y otras. Dentro de la bioinformática, el análisis de secuencias es una de las mayores problemáticas. El análisis de estas secuencias de ADN, ARN y proteínas es uno de los campos más amplios en la bioinformática y puede verse desde diferentes puntos dentro de la ciencia de la computación. Precisamente en este trabajo nos centraremos en el análisis que tiene que ver con predicción de funciones o hipótesis a partir de secuencias biológicas.

En términos de ciencia de la computación los problemas pueden dividirse, teniendo en cuenta si tenemos conocimiento de la función o hipótesis que se desea predecir, en problemas supervisados y no supervisados. Los problemas supervisados son aquellos donde se tiene información de la hipótesis, y los no supervisados donde no se tiene

información de la misma. Los algoritmos relacionados con los problemas supervisados muchas veces son conocidos como clasificadores, así como los que tienen que ver con los problemas no supervisados como métodos de agrupamiento. Este trabajo se concentra en problemas supervisados de análisis de secuencias.

Son muchas las técnicas dentro de la inteligencia artificial que han sido empleadas para resolver diversos problemas de análisis de secuencias dentro de la Bioinformática (Larrañaga, Calvo *et al*., 2006) (Baldi and Soren, 2001). Se destacan el uso de árboles de decisión (Salzberg, Delcher *et al*., 1998) (James, 2004), k vecino más cercano (Kim, 2004), support vector machine (Carter, Dubchak *et al*. 2001) (Krishnapuram, Carin *et al*., 2004). Pero las redes neuronales no se quedan atrás, sobre todo en problemas de clasificación la red conocida como multilayer perceptron ha sido usada con buenos resultados (Draghici and Potter, 2003) (Wang and Larder, 2003). Pero las redes recurrentes también han incursionado en esta temática de la bioinformática (Baldi, Brunak *et al*., 2000) (Boden and Hawkins, 2005) (Bonet García *et al*., 2007). Justamente estos resultados anteriores y otros análisis realizados sobre las potencialidades matemáticas de estas redes neuronales (Hawkins and Boden, 2005) nos hacen llegar a concluir que pueden ser métodos muy eficientes para analizar secuencias biológicas como secuencias temporales.

Los objetivos de este trabajo son:
Demostrar que las redes recurrentes para tratar secuencias pueden ser superiores a otros métodos de inteligencia artificial.
Elaborar una herramienta que permita la construcción de estas redes para utilizarlas en cualquier problema de este tipo en bioinformática o en cualquier otro campo donde se necesite el trabajo con secuencias o series.

Redes Neuronales Recurrentes

Una red neuronal puede ser caracterizada por el modelo de la neurona, el esquema de conexión que presentan sus neuronas, o sea su topología, y el algoritmo de aprendizaje empleado para adaptar su función de cómputo a las necesidades del problema particular.

Existe una amplia variedad de modelos de neuronas, cada uno se corresponde con un tipo determinado de función de activación y de salida de la neurona.

La topología es la forma específica de conexión (arquitectura) y la cantidad de neuronas conectadas (el número de parámetros libres) que describen una red. En los últimos años se ha producido una amplia variedad de topologías de redes neuronales, sin embargo, la mayoría de ellas se encuentran ubicadas en dos grandes grupos: las redes multicapa de alimentación hacia adelante (feed-forward) y las redes recurrentes (RNR).

Las redes feed-forward no tienen ciclos, las neuronas están organizadas en capas que se conectan de manera unidireccional. Generalmente estas redes son denominadas estáticas, pues producen una única salida para un conjunto de entrada, o sea, el estado de una red es independiente del estado anterior.

Por otro lado las redes recurrentes son sistemas dinámicos. El cálculo de una entrada, en un paso, depende del paso anterior y en algunos casos del paso futuro.

Las RNR son capaces de realizar una amplia variedad de tareas computacionales incluyendo el tratamiento de secuencias, la continuación de una trayectoria, la predicción no lineal y la modelación de sistemas dinámicos.

Estas redes también se conocen como redes espacio-temporales o dinámicas, son un intento de establecer una correspondencia entre secuencias de entrada y de salida que no son más que patrones temporales.

Existen tres tipos de tareas esenciales que se pueden realizar con este tipo de redes:
Reconocimiento de secuencias: Se produce un patrón de salida particular cuando se especifica una secuencia de entrada.

Reproducción de secuencias: La red debe ser capaz de generar el resto de una secuencia cuando ve parte de ella.

Asociación temporal: En este caso una secuencia de salida particular se debe producir en respuesta a una secuencia de entrada específica.

Una RNN se puede clasificar en parcial y/o totalmente recurrente. Las totalmente recurrentes son aquellas que cada neurona puede estar conectada a cualquier otra y sus conexiones recurrentes son variables. Las redes parcialmente recurrentes son aquellas que sus conexiones recurrentes son fijas. Estas últimas son la forma usual para reconocer o reproducir secuencias. Generalmente tienen la mayoría de las conexiones hacia delante pero incluyen un conjunto de conexiones retroalimentadas.

Existen varios tipos de redes definidas con su topología y sus algoritmos de aprendizaje.

Algoritmo de aprendizaje
Muchos algoritmos exactos y aproximados han sido desarrollados para el entrenamiento de las redes recurrentes. La mayoría de ellos basan su trabajo en la técnica del gradiente descendente y pueden ser agrupados, según Atiya (Atiya and Parlos, 2000), en 5 clases generales:
Backpropagation Through Time (BPTT),
Forward Propagation o Real Time Recurrent Learning (RTRL),
Fast Forward Propagation,
Funciones de Green
Actualización en Bloque (*Block Update*).

Entre las técnicas de gradiente descendente más empleadas para el entrenamiento de redes recurrentes se encuentran BPTT. El algoritmo BPTT es una variante extendida del Backpropagation (Rumelhart, Hinton *et al*., 1986) original.

BPTT se basa en desplegar la red para cada intervalo de tiempo, convirtiéndola en una red con conexiones hacia delante, con pesos compartidos. Y luego se entrena la red desplegada con el backpropagation, y se calcula el error, conservando los pesos compartidos. Finalmente, el análisis de una nueva secuencia implica también un despliegue de la red. Pearlmutter hace referencia a varias de las aplicaciones de este algoritmo (Pearlmutter, 1990).

Este tipo de redes, tiene por demás, una importancia propia, por sus posibilidades de aplicación en muchas otras esferas. Su grado de "generalidad" desde el punto de vista de

los potenciales de aplicación llega a formular un reto a los especialistas en Ciencias de la Computación en el sentido de facilitar u optimizar la construcción de redes de este tipo.

Redes Recurrentes Bidireccionales Dinámicas

Los modelos conexionistas para aprendizaje en los dominios secuenciales son, usualmente, sistemas dinámicos que necesitan estados ocultos para almacenar información contextual. Estos modelos pueden adaptarse a tiempos variables, en dependencia de las entradas del mismo, lo cual puede permitir el trabajo con secuencias de tamaños diferentes.

Los problemas de aprendizaje de secuencias, todavía hoy en día se exploran, a pesar de las numerosas aplicaciones que se han desarrollado. En particular la traslación secuencial es generalmente una tarea dura y los modelos actuales parecen tener varias limitaciones. Una de estas limitaciones es la suposición de que el sistema es causal.

Un sistema dinámico se denomina causal si la salida en el tiempo t no depende de entradas futuras. Y se denomina no-causal si en la salida del tiempo t se tiene en cuenta la información de los tiempos pasados y futuros. Para ciertas categorías de secuencias finitas, la información del tiempo pasado y futuro es muy importante (Baldi, Brunak *et al*., 2000).

En la Figura 1 se muestra un ejemplo de topología bidireccional. Esta topología consiste de dos bloques de contexto, uno con recurrencia al pasado y otro con recurrencia al futuro. En cada tiempo t, entiéndase como pasado aquellas capas que tienen conexiones que dependen de tiempos menores que t y como futuro, aquellas que dependen de tiempos mayores que t.

Téngase en cuenta que en la Figura 1, cada una de las flechas que van de capa a capa, significan que hay conexión de todas las neuronas de la capa origen con todas las neuronas de la capa destino. Las flechas discontinuas representan las conexiones en el tiempo, el operador de desplazamiento q+1 significa que la conexión viene desde un tiempo inmediato anterior, y el operador q-1 significa que la conexión viene del tiempo inmediato posterior.

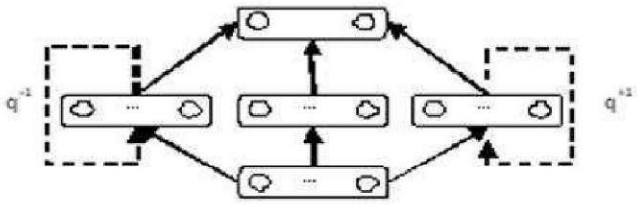

Fig. 1 Topología de una Red Neuronal Bidireccional.

Si se analiza detenidamente este tipo de topología se puede ver que tiene tres capas ocultas que hacen diferentes funciones. Cada una de estas capas realiza una transformación de rasgos basada en pesos distintos. La transformación de la izquierda se le pasa como información al tiempo anterior, la transformación de la derecha al tiempo posterior y la transformación del medio es información para tiempo actual. Esto hace que se puedan analizar problemas donde exista relación entre partes de la secuencia, relaciones que nos tienen que ser iguales. Es precisamente una de las características que presentan las secuencias biológicas, en las que se encuentran dependencias complejas entre los rasgos. Otra de las características que distingue a estos problemas de análisis de secuencias y que hace pensar de nuevo en estas redes es que las regiones de decisión entre las clases son también complejas.

Plataforma para la creación y entrenamiento de redes recurrentes

La gran variedad de redes recurrentes, hace que las topologías sean diversas así como sus mecanismos de aprendizaje. Para poder hacer una plataforma que permita construir redes recurrentes se necesita que ésta tenga una gran flexibilidad.

Existen varios software en la literatura para el tratamiento con redes neuronales. En realidad, hasta ahora han sido creados varios simuladores (SNNS, NeuroSolution, etc.) y lenguajes de especificación (Aspirin, CONNECT, etc.) con este propósito; pero todos presentan limitaciones de flexibilidad y alcance que dificultan el desarrollo sistemático de soluciones a problemas reales. Todos estos software tienen redes fijas, los más comunes usan redes recurrentes de Elman (Elman 1990), pero no permiten usar topologías de nueva creación, como la que pretendemos construir para nuestro trabajo.

En efecto, se hace evidente la necesidad de disponer de un ambiente de desarrollo que brinde la posibilidad de especificación y ejecución de los más diversos modelos de redes neuronales existentes, así como la posibilidad de extensión, combinación y creación de nuevos modelos (sin tener que modificar la herramienta misma) si es requerido, todo ello de manera programática.

El N-Engine v1.0 es un programa para plataforma Windows 9x/2000/2003 construido bajo el diseño de máquina virtual antes expuesto, concebido en su primera versión para el diseño, entrenamiento y validación del modelo de red feed-fordware multicapa y Recurrente basado en Backpropagation Trought Time. Opera en un entorno aislado como un nodo individual, aunque presenta una arquitectura de software para su fácil extensión a entornos distribuidos y a otros modelos de redes diferentes. La plataforma N-Engine implementa el algoritmo de aprendizaje Backpropagationy Backpropagation Trough Time basado en gradiente descendiente. Básicamente esta constituido de tres módulos fundamentales: el núcleo N-Kernel v1.0, la Interfaz gráfica N-GraphConsole v1.0 y el intérprete de lote de comandos N-CBC v1.0 (Neuronal Command Batch Compiler).

N-Kernel v1.0 posee una arquitectura basada en subprocesos dedicados que corren concurrentes dentro de la aplicación cada uno destinado a tareas específicas; con una comunicación interna basada en intercambio de mensajes. Un planificador incorporado basado en un modelo de operadores de exclusión controla la actividad entre los subprocesos, de manera que cada uno no interfiera la actividad de los demás. Exporta un conjunto de rutinas, denominadas genéricamente API, que abstrae la funcionalidad del núcleo hacia el exterior.

La interfaz gráfica N-GraphConsole v1.0 permite a los usuarios la operación del software de una forma cómoda y fácil de aprender. Presenta visores que asisten gráficamente el proceso de diseño, el control del entrenamiento y un editor de texto para el trabajo con los comandos y las bases de casos.

N-CBC v1.0 constituye un intérprete de comando incorporado al sistema capaz de analizar una secuencia de sentencias en el lenguaje NCBL (Neuronal Command Batch Language) y traducirlas a las rutinas correspondiente de la API. Es a través de este intérprete que el usuario se comunica con el sistema y controla su funcionalidad.

El NCBL es un lenguaje sencillo creado para la comunicación del usuario con el núcleo del sistema. Inicialmente presenta forma de lotes de instrucciones atómicas que pueden ir incrementando su repertorio a medida que el sistema aumente su complejidad. Está

constituido por tres bloques fundamentales de instrucciones: sentencias de entrada/salida de información, sentencias para el ajuste de parámetros, sentencias para el diseño de topologías y sentencias para la ejecución de los algoritmos de entrenamiento [1].

Típicamente el trabajo con N-Engine involucra la creación de una topología de red específica para cada problema.

```
// crear topología para red MLP para el problema XOR
Create Layer LayIn as Length 2, double, input
Create Layer LayMd as Length 2, double, middle
Create Layer LayOut as Length 1, double, output
Create Connection Con1 as delay 0, source LayIn, target LayMd
Create Connection Con2 as delay 0, source LayMd, target LayOut
// crear una neurona
Create Neuron Ner1 as binary
```

Las cinco primeras sentencias crean la topología para una red que soluciona el problema del XOR [2] (una capa de entrada LayIn de dos neuronas que manejan información de tipo real, una capa intermedia LayMd con las mismas características que la anterior y una capa de salida con una neurona tipo real). Las tres primeras crean objetos tipo layer y la dos siguientes, de tipo connection. Los objetos tipo layer brindan una abstracción de diseño para el manejo de capas enteras de neuronas, cada una recibe un nombre distinto por el cual puede ser referenciada en próximas sentencias. Los objetos layer poseen una longitud o cantidad de neuronas determinada y un tipo de valor de información que procesa (real en el presente ejemplo), además de un modificador que caracteriza la misma en capa de entrada, intermedia o de salida. Los objetos de tipo connection poseen un nombre de objeto origen y destino que puede ser un objeto tipo layer o neuron, adicionalmente son modificados por un operador delay, característico de los modelos de red recurrentes.

La próxima sentencia muestra la manera de crear una neurona de nombre Ner1 usando el lenguaje NCBL. La creación de un objeto tipo connection que involucre dos layers significará la creación implícita de n*m conexiones, donde n y m significan las cantidades de neuronas de cada una de las capas. Un objeto tipo neuron es tratado como un layer de una sola neurona.

La Figura 2 muestra las ventanas del NEngine a la izquierda la ventana de introducción de código, a la derecha de arriba hacia abajo, la ventana del error de clasificación, el resultado del porciento de casos bien clasificados (medida de exactitud) de la base d entrenamiento y de la base de prueba y por último el modelo gráfico de la red construida.

---

[1] El NCBL comentado en el presente trabajo constituye una versión muy reducida para la operación con el N-Engine v1.0 y es un subconjunto de la versión original de la cual se prevé presente una estructura más sofisticada e incorpore instrucciones para el control de ejecución, manejo de expresiones y la descripción, mediante refinamiento, de nuevos modelo de redes: capacidades previstas en el diseño de la Máquina Virtual original.

[2] El problema XOR es un problema típico en la literatura que describe las redes MLP, consiste en la simulación de la función lógica xor, mediante el uso de una red de una sola capa intermedia.

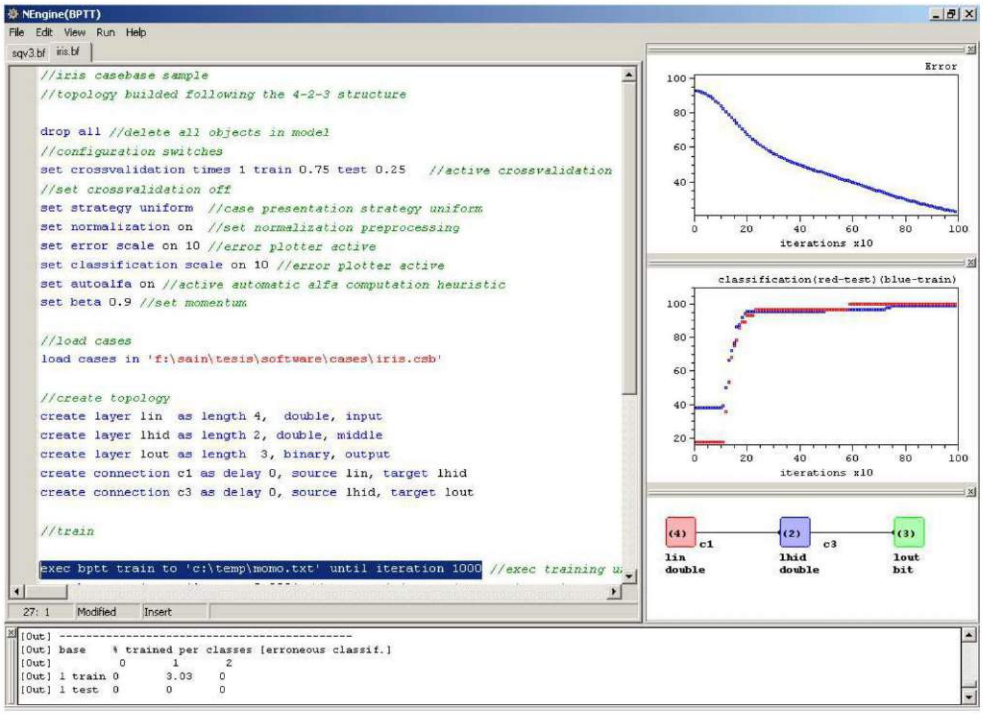

Fig. 2 Ventanas de NEngine.

El software permite crear una red con una topología cualquiera, con enlaces en cualquier dirección y permite la incorporación de otros algoritmos de aprendizaje con facilidad, lo cual lo hace más potente. Tiene las siguientes facilidades:

- Lenguaje de especificación.
- Construcción de distintas topologías.
- Selección de distintas funciones de error y de activación en la capa de salida.
- Uso de crossvalidation para estimar la eficiencia de la red.
- Entrenamiento con el uso de los métodos: BP y BPTT.

La arquitectura, como ya se ha dicho, se basa en una estructura de clases para la gestión de procesos concurrentes, con facilidades para su extensión a una plataforma unificadora de redes neuronales.

Esta red fue utilizada en la clasificación de resistencia del VIH con resultados muy buenos y superiores a los obtenidos anteriormente con otros métodos de inteligencia artificial (Bonet, García *et al*., 2007).

Conclusiones

En este trabajo se obtiene una herramienta para entrenar redes neuronales recurrentes con BTT como algoritmo de aprendizaje., que podrá ser usada para cualquier problema de análisis de secuencias o series en el tiempo. Este software tiene facilidades visuales para analizar el error de entrenamiento y evitar el sobreentrenamiento. Permite el uso de validación cruzada, así como el uso de varias funciones de error y validación.

Se implementa un módulo para redes recurrentes bidireccionales. Se hace un análisis de la utilidad de estas redes para enfrentar sistemas no-causales, en particular problemas de bioinformática relacionados con análisis de secuencias de DNA, RNA o proteínas.

Referencias

Atiya, A. F. and A. G. Parlos "New Results on Recurrent Network Training: Unifying the Algorithms and Accelerating Convergence." IEEE Transactions on Neural Networks, (2000), 11(3): 697.

Baldi, P., S. Brunak, *et al* "Bidirectional Dynamics for Protein Secondary Structure Prediction." In: R. Sun and C.L. Gile (eds.) Sequence Learning, LNAI, (2000), 1828: 80-104.

Baldi, P. and B. Soren. Bioinformatics: The Machine Learning Approach, MIT Press, (2001).

Boden, M. and J. Hawkins."Prediction of subcellular localisation using sequence biased recurrent networks." Bioinformatics, (2005): 2279-2286.

Bonet, I., M. M. García, *et al*. Predicting Human Immunodeficiency Virus Drug Resistance Using Recurrent Neural Networks., Springer-Verlag Berlin Heidelberg, (2007).

Carter, R. J., I. Dubchak, *et al*."A computational approach to identify genes for functional RNAs in genomic sequence." Nucleic Acids Research, (2001), 29(19): 3928-3938.

Draghici, S. and R. B. Potter "Predicting HIV drug resistance with neural networks." Bioinformatics, (2003), 19(1): 98-107.

Elman, J. L. "Finding structure in time." Cognitive Science, (1990), 14(2): 179-211.

Hawkins, J. and M. Boden. "The Applicability of Recurrent Neural Networks for Biological Sequence Analysis." IEEE/ACM Transactions on Computational Biology and Bioinformatics, (2005), 2(3): 243-253.

James, R. Predicting Human Immunodeficiency Virus Type 1 Drug Resistance from Genotype Using Machine Learning, University of Edinburgh, (2004).

Kim, S. "Protein b-turn prediction using nearest-neighbor method." Bioinformatics, (2004), 20(1): 40-44.

Krishnapuram, B., L. Carin, *et al* "Joint classifier and feature optimization for comprehensive cancer diagnosis using gene expression data." Journal of Computational Biology, (2004), 11(2-3): 227-242.

Larrañaga, P., B. Calvo, *et al*. "Machine learning in bioinformatics." Briefings in Bioinformatics, (2006), 7(1): 86-112.

Pearlmutter, B "Dynamic Recurrent Neural Networks." DARPA Research, (1990).

Rumelhart, D. E., A. G. E. Hinton, et al. Learning internal representations by error propagation Parallel distributed processing: explorations in the microstructure of cognition, MIT Press, (1986), vol. 1: foundations 318-362.

Salzberg, S., A. L. Delcher, *et al*."A decision tree system for finding genes in DNA." Journal of computational biology : a journal of computational molecular cell biology, (1998), 5(4): 667-680.

Wang, D. C. and B. Larder. "Enhanced prediction of lopinavir resistance from genotype by use of artificial neural networks." Journal Of Infectious Diseases, (2003), 188(5): 653-660.