**RED NEURONALES RECURRENTES**

**Nombre:** Santiago Andres Del Valle Pinilla

**ID:** 00042774

**¿Qué es una RNR?**

“Una red neuronal es un modelo simplificado de cómo el cerebro humano procesa la información: funciona conectando simultáneamente una gran cantidad de unidades de procesamiento interconectadas que parecen versiones abstractas de neuronas”-IBM

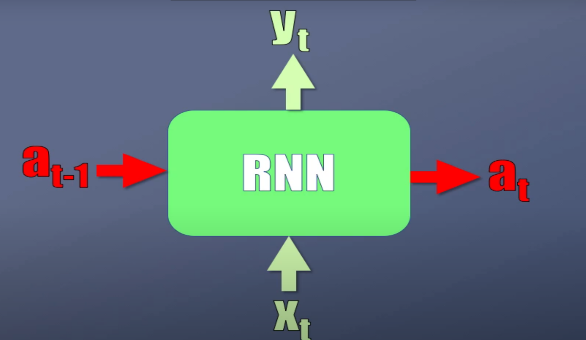
En resumen, una red neuronal simple son algoritmos los cuales tienen por características ser adaptivos referentes al aprendizaje, tolerantes a los cambios/errores y operativos en tiempo real; Es un tipo de Inteligencia artificial débil la cual se entrena mediante generaciones con el fin de especializarse en un área de despliegue.

Las RNR son un tipo de red neuronal recursiva la cual permite que dos neuronas puedan interaccionar y transmitirse información recíprocamente, en lugar de unidireccionalmente, formando con ello bucles de retroalimentación de esta manera siendo muy útiles para áreas especificas como el procesamiento del lenguaje natural.

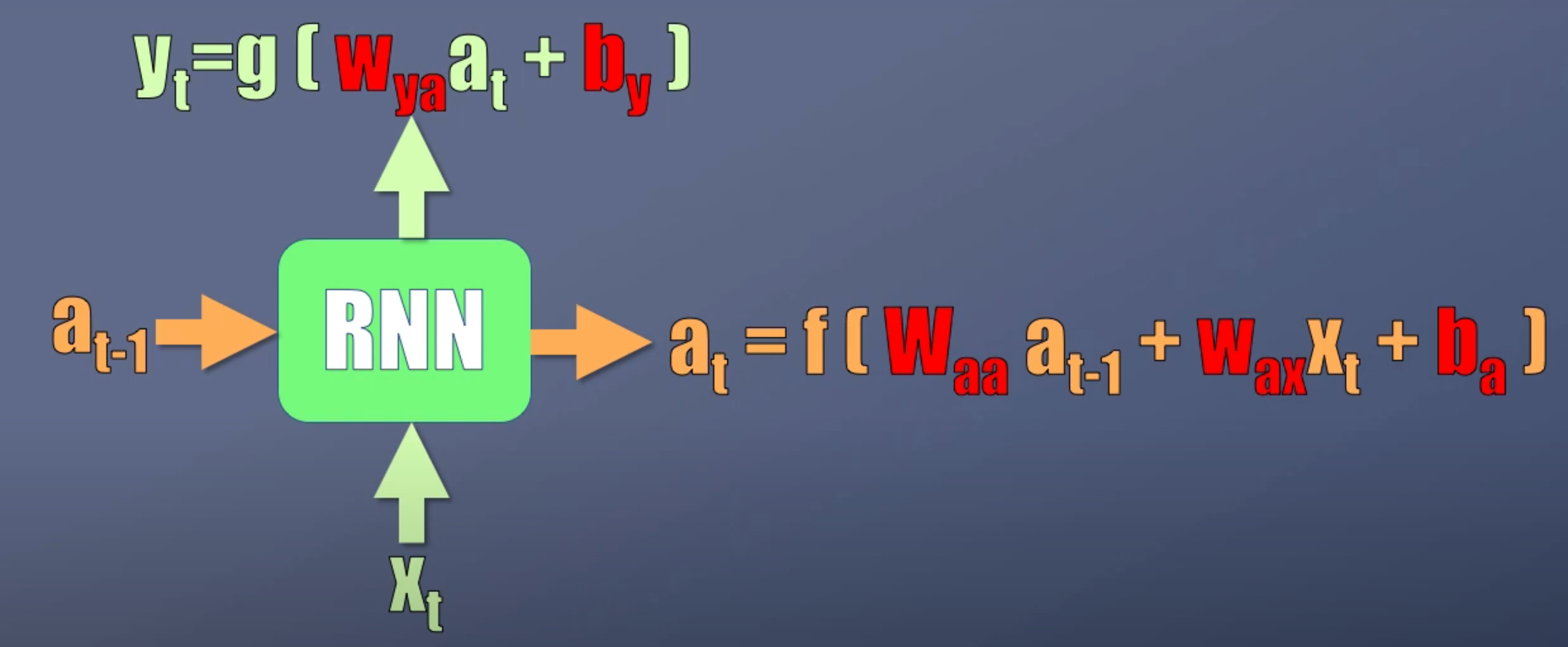
**Analogía Humana**

Científicamente el cerebro humano se comunica entre sí mediante las neuronas, estas mismas a la vez se comunican entre si varias veces mediante sus componentes físicos , de esta manera se logra una perfecta recepción y respuesta a los estímulos externos por lo cual Warren McCulloch y Walter Pitts en 1943 notaron la existencia de circuitos de neuronas creando así la memoria temporal.

**Funcionamiento**

Una red neuronal recurrente es capaz de permitir que una secuencia de vectores de entrada forme una secuencia de vectores de salida además de la entrada predefinida, es decir, mientras que otros tipos de redes tienen un dato de entrada, esta tiene dos.

Como se puede apreciar en la imagen, en cada instante de tiempo (En el caso del procesamiento de lenguaje natural, este instante de tiempo se refiere a cada carácter) se procesan dos salidas mediante dos entradas, at-1 el resultado del procesamiento de la anterior neurona y xt el input , así se comparte y se preserva la información entre varios instantes de tiempos. Calcular la salida y activación de la red recurrente a partir de sus dos entradas se utiliza la misma lógica que las neuronas artificiales tradicionales. En la anterior imagen se aprecia que la neurona artificial tiene una entrada "x" y produce la salida "y", la salida es un par datos de entrada: transformación y función de activación no lineal. En el caso de redes recurrentes, las activaciones se calculan de manera similar, y transforma el primer resultado de los datos de entrada y luego llevarlos a una función de aproximación. Esto se perfecciona mediante el entrenamiento.



*Esto en cada instante de tiempo*

**Explicación del código**

El código anexo el cual hace uso de estos conceptos, consiste en la generación de nombres aleatorios utilizando una base de datos predefinida con el fin de que el modelo aprenda de patrones y genere nombres más exactos referente a los predispuestos.

Este código posee 4 módulos los cuales son:

1. **Análisis y limpieza de datos:** En este módulo se debe limpiar/filtrar los datos con el fin de posteriormente utilizarlo en el modelo de Keras , en este caso , se utilizo un dataset de nombres de personas que sin embargo , poseían caracteres no deseados por el cual se tuvo que elegir una muestra y de esa muestra poder eliminar dichos caracteres para posteriormente generar dos diccionarios que puedan servir de “instantes de tiempo en el sistema”.
2. **Implementación en Keras:** Se definen parámetros en la librería de Kera (Como las neuronas, celdas de entrada y salida, etc) con el fin de posteriormente ser utilizado en el entrenamiento “supervisado”.
3. **Entrenamiento supervisado:** En este caso se toman nombres aleatorios y se empieza a generar la función del modelamiento de la palabra, es decir, se ejecuta un modelo de entrenamiento el cual sirve como plantilla y guía al sistema para poder ejecutar el resultado.
4. **Entrenamiento:** Se ejecuta el entrenamiento de la RNR con parámetros predefinidos como la cantidad de iteraciones/generaciones para ejecutar el proceso y la cantidad de pruebas por iteración para pasar a la otra generación.
5. **Impresión:** Cuando el modelo termina su ciclo , se crea una función la cual permite imprimir 100 nombres parecidos o basados en la base de datos.

**Resultados Código**



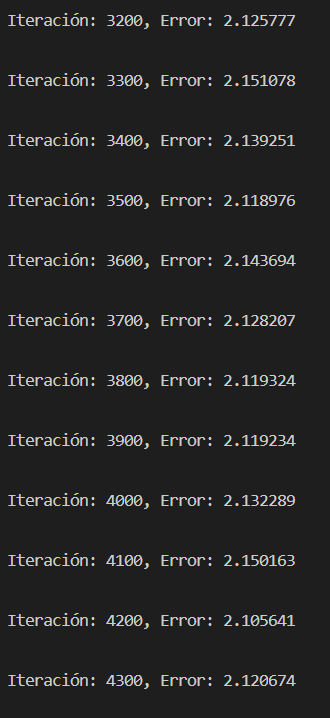
Se establecieron los siguientes parámetros los cuales son

* BATCH\_SIZE = 500 PRUEBAS POR GENERACIÓN
* NITS = 5000 GENERACIONES

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

El programa periódicamente te va comentando el número de iteración (100 en 100) con su respectivo error.



Se puede apreciar que desde la iteración 3000 el porcentaje de error ha variado por lo cual es un buen momento para parar.

Cuando se llegan a las 5000 Generaciones (Se tomo 30 minutos con un I5 de onceava generación) el programa se para y se ejecuta la función de “imprimir” para poder ver los nombres que genero; Como se puede ver que, aunque no son 100% idénticos ya se hace una idea a lo que se asimilo con tan solo este rango de ejecución.

Texto

Descripción generada automáticamente