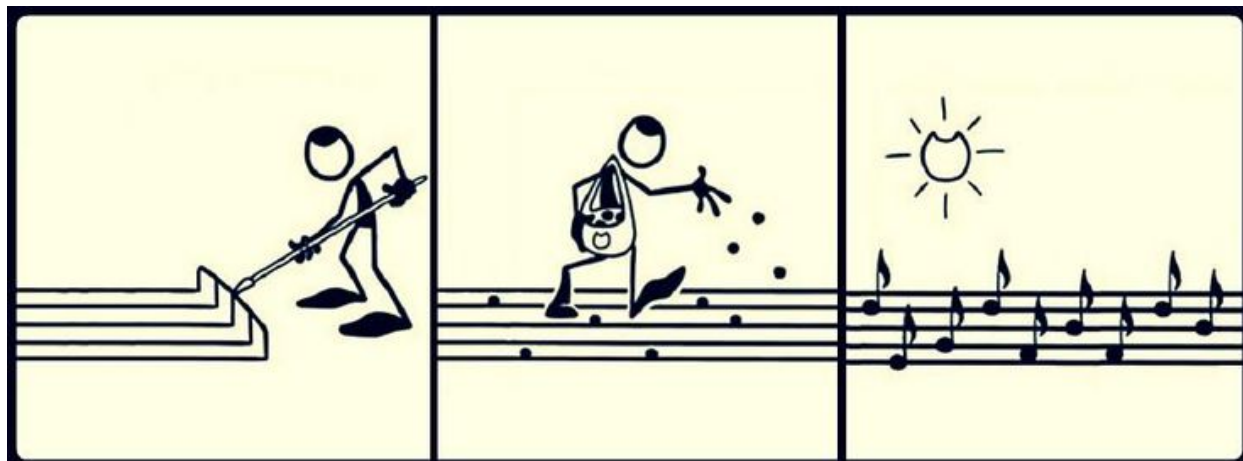


Generador de música

Proyecto final de la asignatura de Sistemas Inteligentes del itinerario de Computación de Ingeniería Informática de la Universidad de La Laguna



Nombre	Contacto
Erik Andreas Barreto de Vera	alu0100774054@ull.edu.es
Jorge Alonso Hernández	alu0100767803@ull.edu.es

1. Resumen

El objetivo del generador de música es diseñar un sistema de aprendizaje basado en muestras. El sistema consta de una base de datos musical que forman parte de la entrada del sistema, a partir de esta información de entrada el sistema es capaz de aprender qué notas musicales son buenas para producir música parecida a la de entrada. De esta forma el sistema es capaz de producir música.

2. Estado del arte

1. Magenta

¿Qué es Magenta?

Magenta es un proyecto del equipo de Google Brain. Magenta tiene dos objetivos.

En primer lugar, es un proyecto de investigación para avanzar el estado del arte en la inteligencia de la máquina para la música y la generación del arte. El aprendizaje automático ya se ha utilizado ampliamente para comprender el contenido, como en el reconocimiento del habla o la traducción.

En segundo lugar, Magenta es un intento de construir una comunidad de artistas, codificadores y investigadores de aprendizaje automático. El equipo principal de Magenta construirá una infraestructura de código abierto alrededor de TensorFlow para hacer arte y música. Comenzaremos con soporte de audio y video, herramientas para trabajar con formatos como MIDI y plataformas que ayudan a los artistas a conectarse a modelos de aprendizaje automático. Por ejemplo, queremos que sea super simple para reproducir música junto con un modelo de rendimiento Magenta.

Evaluación de los resultados del sistema

Evaluar el resultado de los modelos generados es difícil. Llegará el momento en que Magenta tiene 20 modelos diferentes de generación de música disponibles en código abierto. ¿Cómo decidimos cuáles son buenas? Una opción es comparar el resultado del modelo con los datos de entrenamiento midiendo la verosimilitud.

Posibles mejoras del sistema

Para la música y el arte, esto no funciona muy bien. Es fácil generar resultados que están cerca en términos de probabilidad, pero lejos en términos de atractivo (y viceversa). Esto motiva el trabajo en adversarios artificiales como Generar redes neuronales diferentes. Al final, para responder a la pregunta de evaluación necesitamos que las herramientas de Magenta estén en manos de artistas y músicos, y *Magenta media* disponible para los espectadores y oyentes.

Enlace al proyecto

<https://magenta.tensorflow.org/welcome-to-magenta>

3. Introducción

Este sistema generador de música es un sistema inteligente capaz de generar música aprendiendo de diferentes muestras. A medida que el sistema se nutre de una mayor cantidad de muestras el sistema consigue mejorar sus resultados en términos de probabilidad. Esto es diferente a los resultados que nosotros esperemos, más adelante se comentará. A continuación se procede a explicar todo el proceso de creación, estructura, conocimientos, etc que compone este proyecto.

4. Métodos

En este apartado vamos a explicar aquellos conocimientos que hacen falta para comprender el sistema. No vamos a entrar a nivel de matemáticas ni estadísticas sino a nivel de herramientas o modelos.

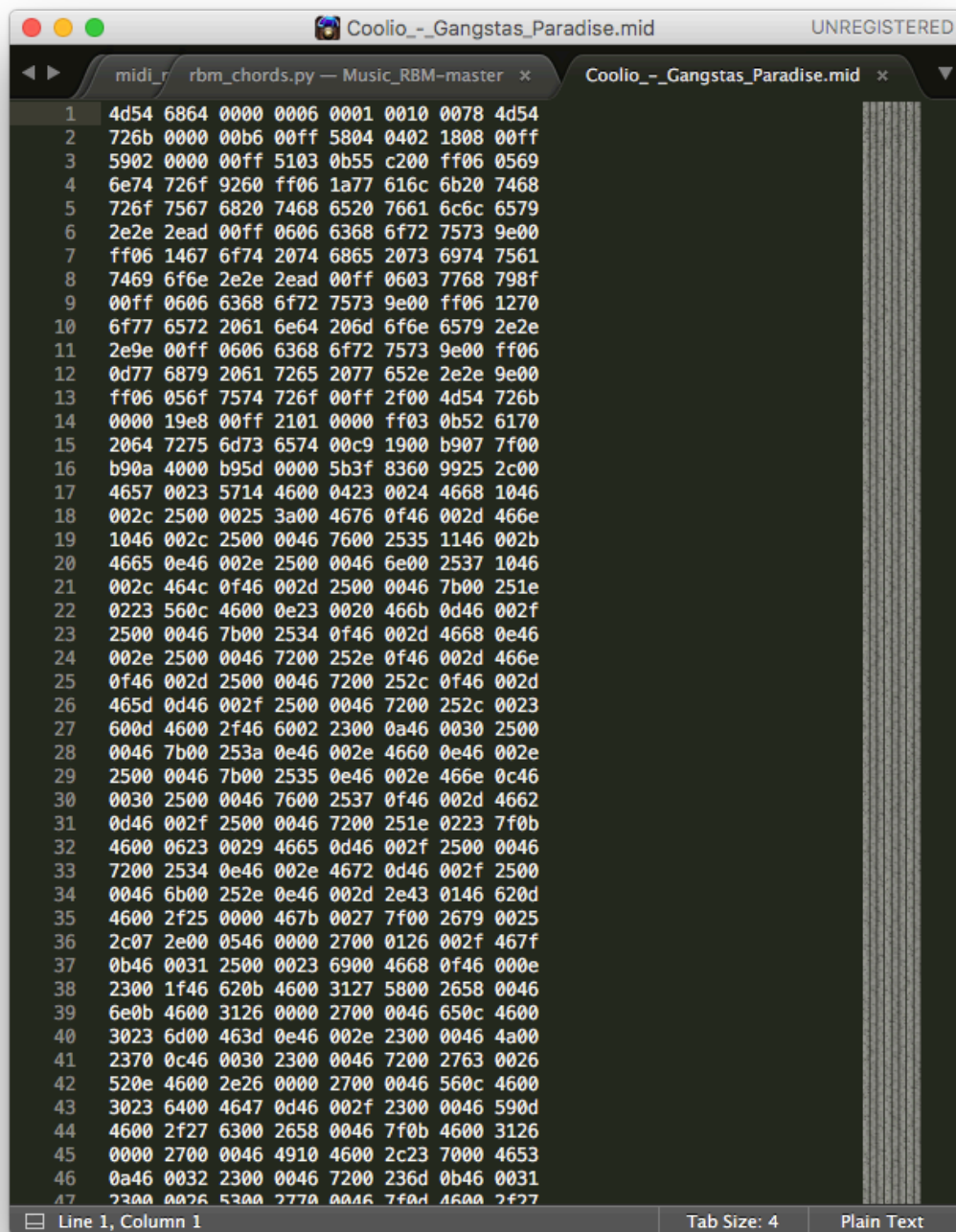
MIDI

¿Qué es MIDI?

MIDI (abreviatura de Musical Instrument Digital Interface) es un estándar tecnológico que describe un protocolo, una interfaz digital y conectores que permiten que varios instrumentos musicales electrónicos, computadoras y otros dispositivos relacionados se conecten y comuniquen entre sí.

¿Qué es un archivo .midi?

El formato estándar MIDI (SMF) permite una manera estandarizada para almacenar, transportar y abrir secuencias en otros sistemas. El compacto tamaño de estos archivos ha permitido que sean implementados de manera numerosa en computadoras. Fueron creados para su uso universal e incluir información como el valor de las notas, tiempo y nombre de las pistas. La lírica puede ser incluida como metadata. Un archivo *.midi* viene codificado de la siguiente forma:



The image shows a screenshot of a MIDI file viewer application. The title bar indicates the file is 'Coolio_-_Gangstas_Paradise.mid' and it is 'UNREGISTERED'. The main window displays a list of MIDI events, numbered 1 through 47. Each event is represented by a line of hexadecimal data. The events include various MIDI commands such as '4d54', '726b', '5902', '6e74', '726f', '2e2e', 'ff06', '1467', '6f74', '2074', '6865', '2073', '6974', '7561', '7469', '6f6e', '2e2e', '2ead', '00ff', '0606', '6368', '6f72', '7573', '9e00', '00ff', '0606', '6368', '6f72', '7573', '9e00', 'ff06', '1270', '6f77', '6572', '2061', '6e64', '206d', '6f6e', '6579', '2e2e', '2e9e', '00ff', '0606', '6368', '6f72', '7573', '9e00', 'ff06', '0d77', '6879', '2061', '7265', '2077', '652e', '2e2e', '9e00', 'ff06', '056f', '7574', '726f', '00ff', '2f00', '4d54', '726b', '0000', '19e8', '00ff', '2101', '0000', 'ff03', '0b52', '6170', '2064', '7275', '6d73', '6574', '00c9', '1900', 'b907', '7f00', 'b90a', '4000', 'b95d', '0000', '5b3f', '8360', '9925', '2c00', '4657', '0023', '5714', '4600', '0423', '0024', '4668', '1046', '002c', '2500', '0025', '3a00', '4676', '0f46', '002d', '466e', '1046', '002c', '2500', '0046', '7600', '2535', '1146', '002b', '4665', '0e46', '002e', '2500', '0046', '6e00', '2537', '1046', '002c', '464c', '0f46', '002d', '2500', '0046', '7b00', '251e', '0223', '560c', '4600', '0e23', '0020', '466b', '0d46', '002f', '2500', '0046', '7b00', '2534', '0f46', '002d', '4668', '0e46', '002e', '2500', '0046', '7200', '252e', '0f46', '002d', '466e', '0f46', '002d', '2500', '0046', '7200', '252c', '0f46', '002d', '465d', '0d46', '002f', '2500', '0046', '7200', '252c', '0023', '600d', '4600', '2f46', '6002', '2300', '0a46', '0030', '2500', '0046', '7b00', '253a', '0e46', '002e', '4660', '0e46', '002e', '2500', '0046', '7b00', '2535', '0e46', '002e', '466e', '0c46', '0030', '2500', '0046', '7600', '2537', '0f46', '002d', '4662', '0d46', '002f', '2500', '0046', '7200', '251e', '0223', '7f0b', '4600', '0623', '0029', '4665', '0d46', '002f', '2500', '0046', '7200', '2534', '0e46', '002e', '4672', '0d46', '002f', '2500', '0046', '6b00', '252e', '0e46', '002d', '2e43', '0146', '620d', '4600', '2f25', '0000', '467b', '0027', '7f00', '2679', '0025', '2c07', '2e00', '0546', '0000', '2700', '0126', '002f', '467f', '0b46', '0031', '2500', '0023', '6900', '4668', '0f46', '000e', '2300', '1f46', '620b', '4600', '3127', '5800', '2658', '0046', '6e0b', '4600', '3126', '0000', '2700', '0046', '650c', '4600', '3023', '6d00', '463d', '0e46', '002e', '2300', '0046', '4a00', '2370', '0c46', '0030', '2300', '0046', '7200', '2763', '0026', '520e', '4600', '2e26', '0000', '2700', '0046', '560c', '4600', '3023', '6400', '4647', '0d46', '002f', '2300', '0046', '590d', '4600', '2f27', '6300', '2658', '0046', '7f0b', '4600', '3126', '0000', '2700', '0046', '4910', '4600', '2c23', '7000', '4653', '0a46', '0032', '2300', '0046', '7200', '236d', '0b46', '0031', '2300', '0026', '5300', '2770', '0046', '7f0d', '4600', '2f27'.

Pero esto a nosotros no nos interesa puesto que usaremos una librería que nos permite manipular este tipo de archivos de una forma sencilla.

¿Por qué MIDI en el proyecto?

Será el formato en el que vamos a trabajar para generar la música. La base de muestras del sistema se encuentra en formato *.midi*, compone la base de entrenamiento.

Más información

En estos enlaces se puede encontrar más información acerca de el formato de notación musical *.midi*:

<http://www.css-audiovisual.com/areas/guias/midi-mensajes.htm>

<https://es.wikipedia.org/wiki/MIDI>

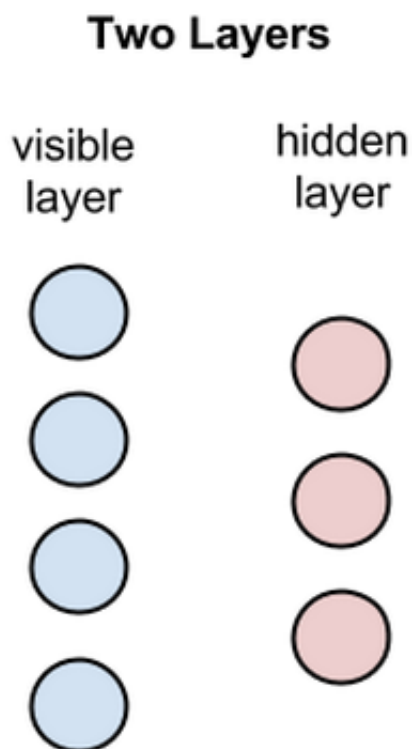
Base de datos musicales

<http://www.midiworld.com>

Restricted Boltzmann machine

¿Qué es? Definición y estructura

Inventado por Geoff Hinton, las Máquinas Boltzmann Restringidas (RBMs) son redes neurales superficiales de dos capas que constituyen los bloques de construcción de redes de creencias profundas. La primera capa del RBM se denomina capa visible, o input, y la segunda es la capa oculta.



Cada círculo en el gráfico anterior representa una unidad similar a una neurona llamada nodo, y los nodos son simplemente donde los cálculos tienen lugar. Los nodos están conectados entre sí a través de capas, pero no hay dos nodos de la misma capa conectados.

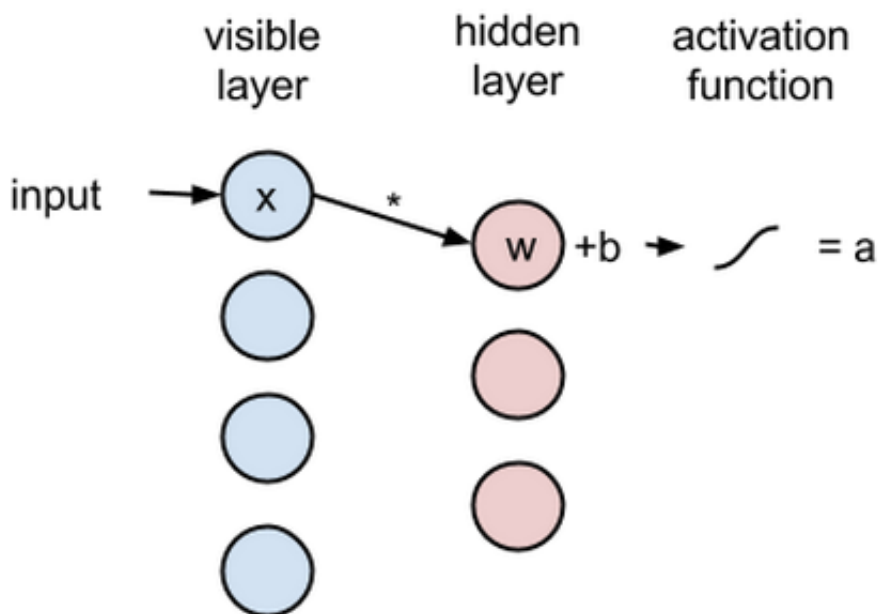
Es decir, no hay comunicación intra-capa - esta es la restricción en una máquina Boltzmann restringida. Cada nodo es un locus de computación que procesa la entrada, y comienza haciendo decisiones estocásticas sobre si transmitir esa entrada o no. (Estocástico significa "determinado al azar", y en este caso, los coeficientes que modifican las entradas se inicializan al azar).

Cada nodo visible toma una característica de bajo nivel de un elemento del conjunto de datos que se va a aprender. Por ejemplo, a partir de un conjunto de datos de imágenes en escala de grises, cada nodo visible recibiría un valor de píxel para cada píxel en una imagen. (Las imágenes MNIST tienen 784 píxeles, por lo que las redes neuronales que las procesan deben tener 784 nodos de entrada en la capa visible).

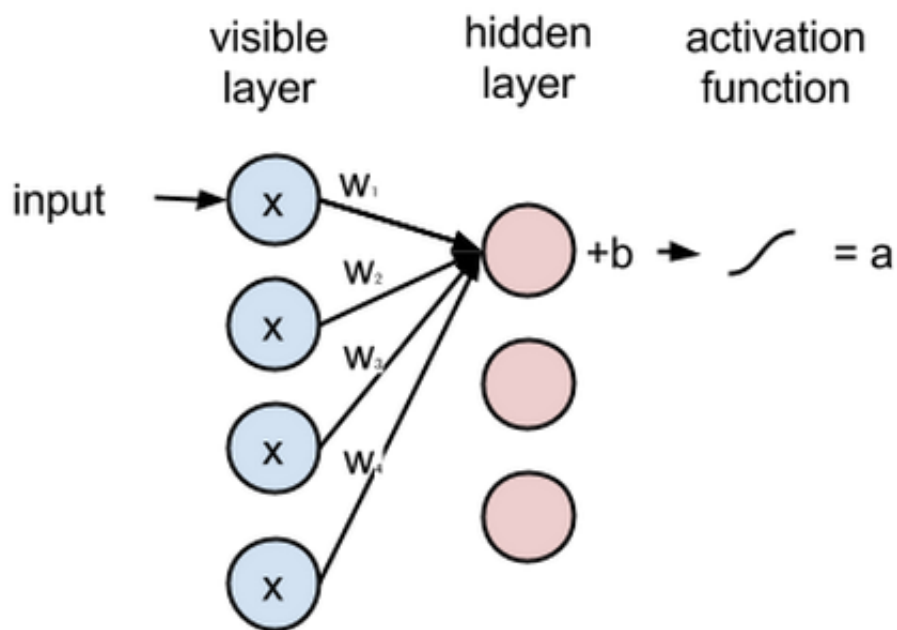
Ahora vamos a seguir ese valor de píxel único, x , a través de la red de dos capas. En el nodo 1 de la capa oculta, x se multiplica por un peso y se añade a un llamado sesgo. El resultado de estas dos operaciones se alimenta a una función de activación, que produce la salida del nodo, o la intensidad de la señal que pasa a través de ella, dada la entrada x .

```
1 | activation f((weight w * input x) + bias b ) = output a
```

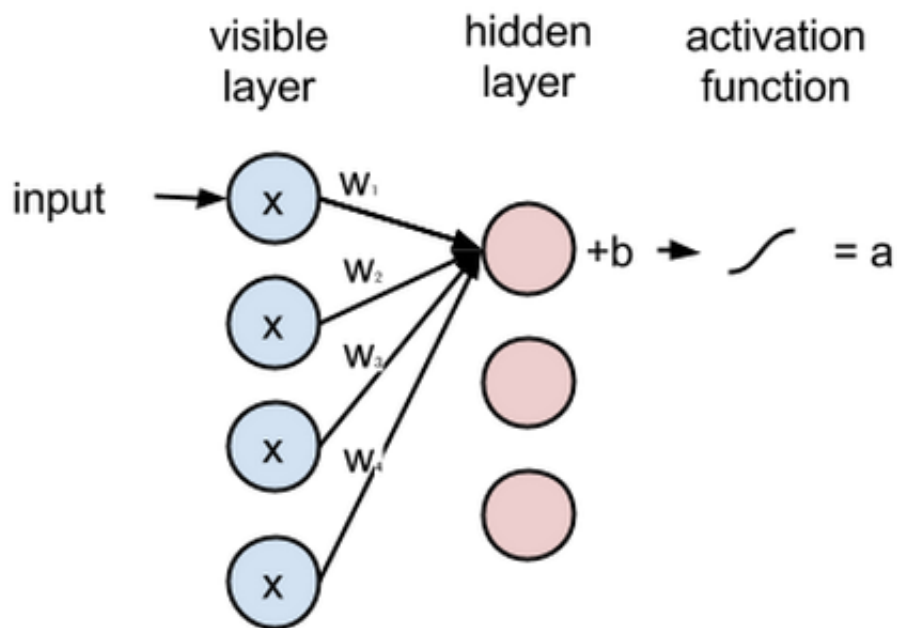
One Input Path



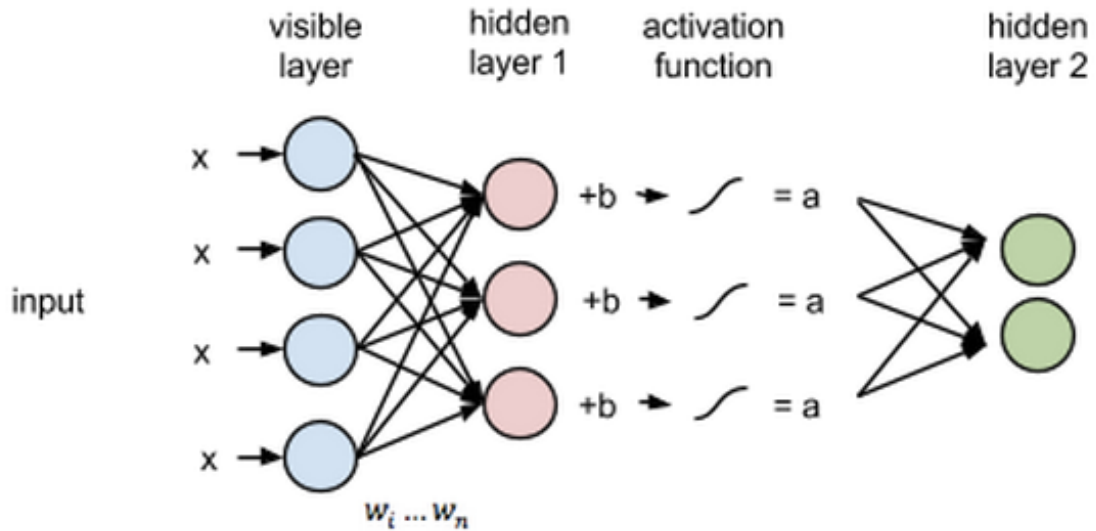
Weighted Inputs Combine @Hidden Node



Weighted Inputs Combine @Hidden Node



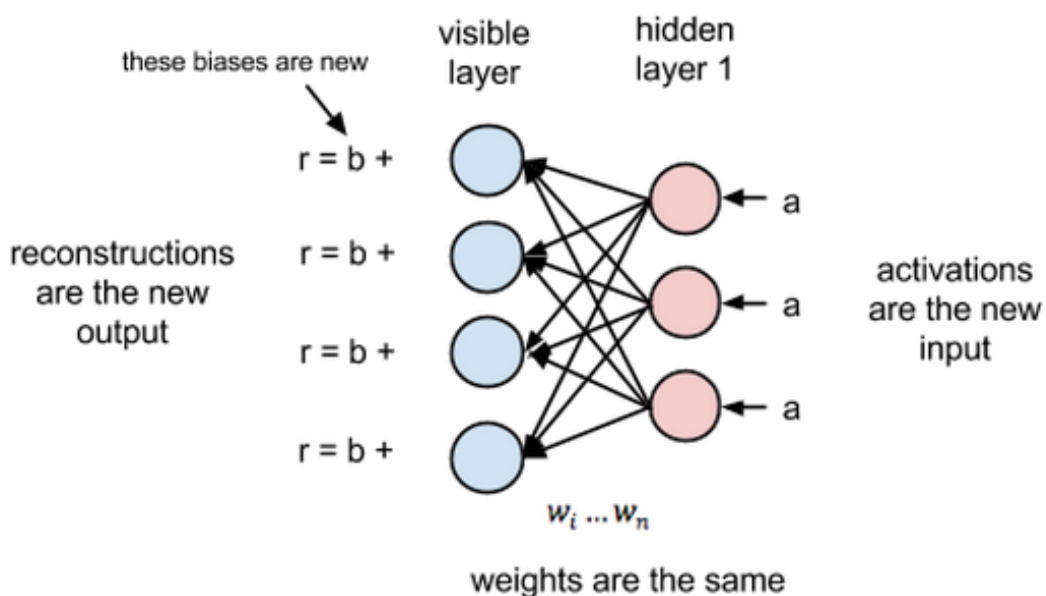
Multiple Hidden Layers



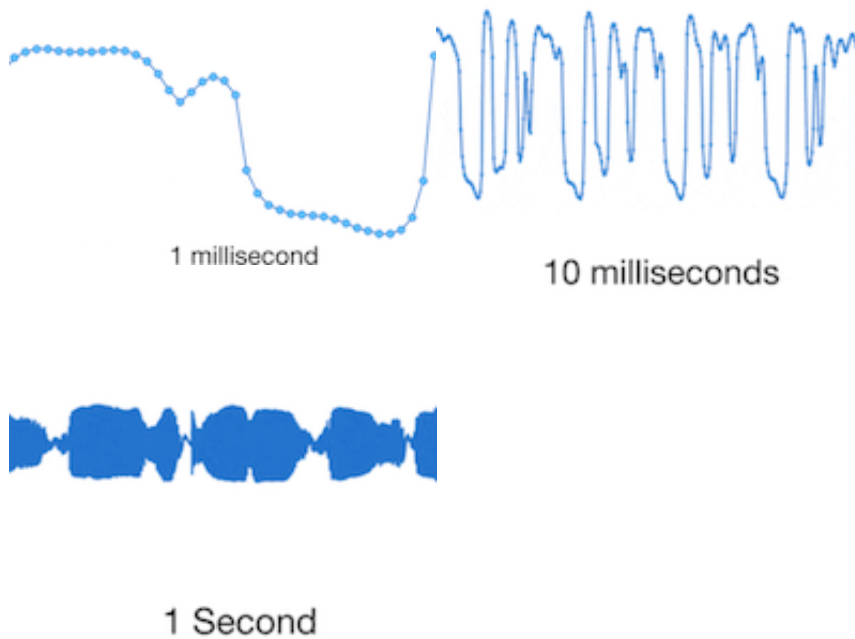
Pero en esta introducción a las máquinas Boltzmann restringidas, nos centraremos en cómo aprenden a reconstruir los datos por sí mismos de una manera no supervisada (medios sin supervisión sin etiquetas de verdad en un conjunto de prueba), haciendo varios pasos hacia adelante y hacia atrás entre la capa visible Y la capa oculta sin implicar una red más profunda.

En la fase de reconstrucción, las activaciones de la capa oculta se convierten en la entrada en un paso hacia atrás. Se multiplican por los mismos pesos, uno por borde de entrenado, así como x se ajustó en peso en el paso hacia adelante. La suma de esos productos se añade a un sesgo de capa visible en cada nodo visible, y la salida de esas operaciones es una reconstrucción; Es decir, una aproximación de la entrada original. Esto puede representarse mediante el siguiente diagrama:

Reconstruction



Generación de cada nota



Enlace a un tutorial para más información

<https://deeplearning4j.org/restrictedboltzmannmachine.html#>

5. Herramientas

◦ Python

Se trata de un lenguaje de programación multiparadigma, ya que soporta orientación a objetos, programación imperativa y, en menor medida, programación funcional. Es un lenguaje interpretado, usa tipado dinámico y es multiplataforma.

¿Por qué python?

Porque TensorFlow es una API sólo para python.

Instalación

[descarga: https://www.python.org/downloads/](https://www.python.org/downloads/)

▪ Pip

pip Es un sistema de gestión de paquetes utilizado para instalar y administrar paquetes de software escritos en Python.

Instalación

A partir de la versión 2.6 pip viene instalado junto con la versión de python.

▪ dependecies

Todos los paquetes que aparecen a continuación se instalan mediante el comando pip que vimos anteriormente.

```
1 | $ pip install
```


■ pandas

Pandas es una librería de python destinada al análisis de datos, que proporciona unas estructuras de datos flexibles y que permiten trabajar con ellos de forma muy eficiente. Pandas ofrece las siguientes estructuras de datos:

- **Series:** Son arrays unidimensionales con indexación (arrays con índice o etiquetados), similar a los diccionarios. Pueden generarse a partir de diccionarios o de listas.
- **DataFrame:** Son estructuras de datos similares a las tablas de bases de datos relacionales como SQL.
- **Panel, Panel4D y PanelND:** Estas estructuras de datos permiten trabajar con más de dos dimensiones. Dado que es algo complejo y poco utilizado trabajar con arrays de más de dos dimensiones no trataremos los paneles en estos tutoriales de introducción a Pandas.

Instalación

```
1 | $ pip install pandas
```

■ numpy

NumPy es una extensión de Python, que le agrega mayor soporte para vectores y matrices, constituyendo una biblioteca de funciones matemáticas de alto nivel para operar con esos vectores o matrices.

Instalación

```
1 | $ pip install numpy
```

■ msgpack

MessagePack es un formato de serialización binaria rápido y compacto, adecuado para datos similares a JSON. Este paquete proporciona enlaces CPython para leer y escribir datos de MessagePack.

Instalación

```
1 | $ pip install msgpack-python
```

■ glob

El módulo glob encuentra todos los pathnames que coinciden con un patrón especificado de acuerdo con las reglas utilizadas por el shell de Unix, aunque los resultados se devuelven en orden arbitrario. Esto facilita la lectura de la base de datos musical.

Instalación

```
1 | $ pip install glob2
```

- **tqdm**

Permite usar barras de progreso en una tarea en la línea de comandos.

```
1 | $ pip install tqdm
```

• TensorFlow

Es una librería open-source dedicada para machine learning.

Instalación

En el siguiente enlace se muestra la instalación en cada sistema operativo.

https://www.tensorflow.org/versions/r0.10/get_started/os_setup

• Github

GitHub es una plataforma de desarrollo colaborativo para alojar proyectos utilizando el sistema de control de versiones Git.

- **Bitbucket**

Bitbucket es un servicio de alojamiento basado en web, para los proyectos que utilizan el sistema de control de revisiones Mercurial y Git.

- **HipChat**

HipChat es un servicio Web para chat interno / privado y mensajería instantánea. Además de un chat individual o en grupo / tema, también ofrece almacenamiento en archivos basado en la nube, videollamadas, historial de mensajes de búsqueda y visualización de imágenes en línea.

6. Entrenamiento

Una vez tenemos el sistema terminado y una base de datos musical, ejecutamos el programa principal para que comience el proceso de entrenamiento. Hemos usado una muestra de 126 canciones, es una muestra pequeña, es por ello que los resultados puedan no ser lo mas ajustados posibles. Para incrementar el porcentaje de acierto se recomienda subir el numero de canciones de muestra. Cómo demuestran muchos otros proyectos, este tipo de sistema generadores de arte tienen la necesidad de muestras mucho mayores, para un generador de música una base de 2000 muestras sería el mínimo recomendable.

Obviamente el entrenamiento con una muestra tan grande incrementa mucho el tiempo de ejecución. Este sistema a partir de la muestra de 126 canciones genera una salida de 10 resultados.

Una vez conocemos este detalle vamos a explicar el proceso:

Primero ejecutamos el comando:

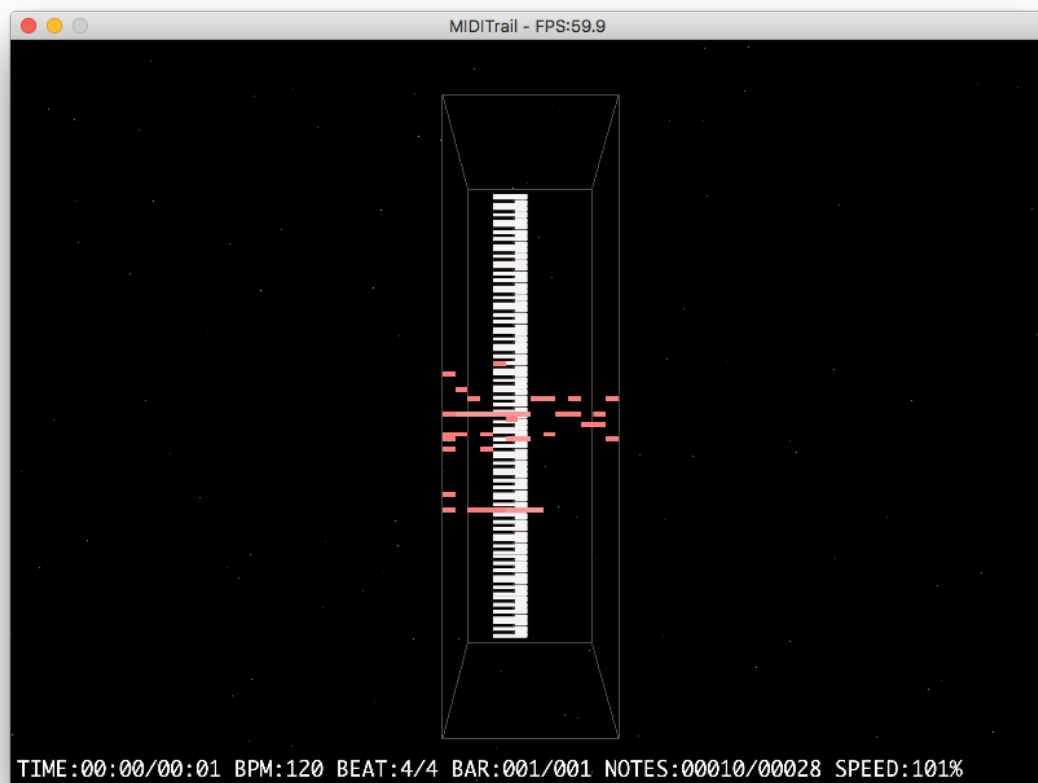
```
1 $ python rbm_chords.py
```

```
[MacBook-Pro-de-Erik:src zephyr$ python rbm_chords.py  
100%|███████████| 126/126 [00:07<00:00, 16.17it/s]  
122 songs processed  
I tensorflow/core/common_runtime/local_device.cc:40] Local device intra op parallelis  
m threads: 2  
I tensorflow/core/common_runtime/direct_session.cc:58] Direct session inter op parall  
elism threads: 2  
   0%|          | 0/200 [00:00<?, ?it/s]  
rbm_chords.py:116: VisibleDeprecationWarning: using a non-integer number instead of a  
n integer will result in an error in the future  
song = song[:np.floor(song.shape[0]/num_timesteps)*num_timesteps]  
   6%|■■■       | 11/200 [00:08<02:17, 1.38it/s]
```

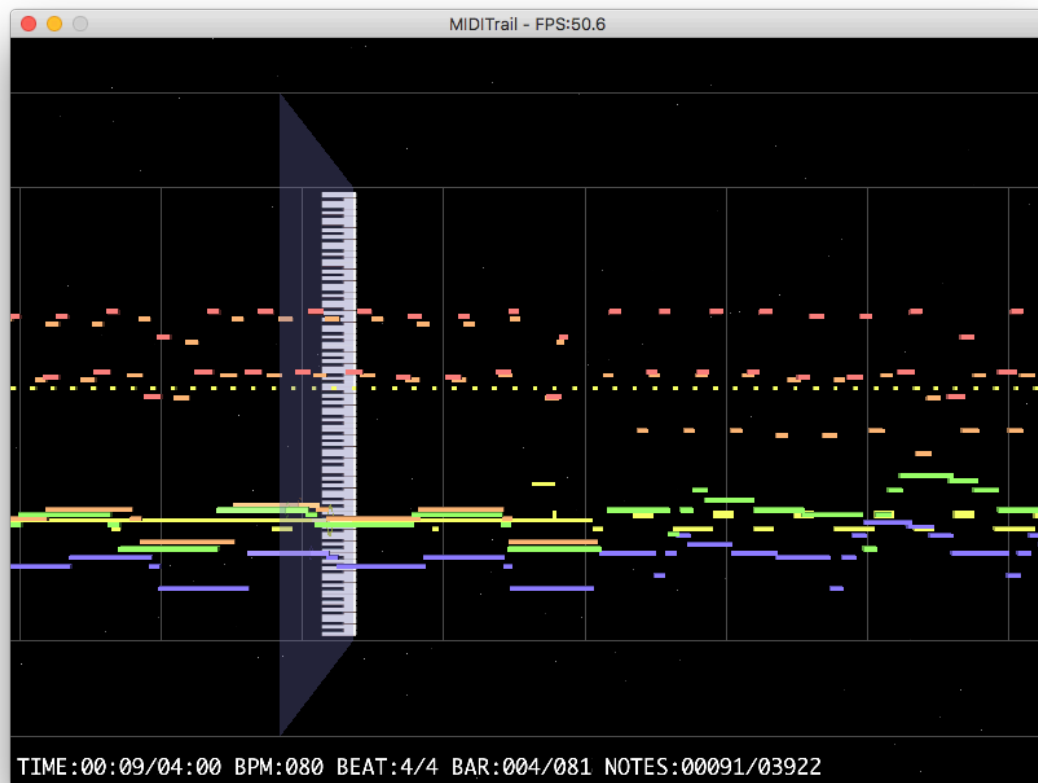
Como vemos el tiempo aproximado de una muestra de 126 canciones es de 2 minutos 17 segundo aproximadamente en un MacBook Pro modelo: *MacBook Pro (13-inch, Mid 2010) Processor 2,4 GHz Intel Core 2 Duo*

7. Resultados

Una vez finaliza la etapa de entrenamiento se genera 10 canciones en el mismo directorio del fichero. Para la reproducción de estos archivos generados hace falta un software específico capaz de reproducir este tipo de formato, no todos los reproductores de música son capaz nosotros hemos usado un reproductor multiplataforma llamado *MIDItail*. Podemos verlo a continuación:



Vamos a destacar que como dijimos al principio del informe el generador sólo genera notas musicales para un mismo instrumento veamos un ejemplo de canción que no es capaz de generar. A continuación aparece una canción famosa de hip hop cuya melodía esta generada por muchas melodías al mismo tiempo(Cada uno corresponde un color:



8. Conclusiones

Con este sistema generador de música hemos sido capaz de generar música de una forma sencilla, lo que hacerlo a mano puede ser mucho más complicado, el contenido del arte tiene el principal problema de la inspiración, se necesitan muchas horas para descubrir nuevas melodías e incluso melodías que nos puedan gustar y de esta forma podemos crear melodías interesantes de una forma sencilla y rápida sin mucho esfuerzo y crear un contenido similar al que queremos y hemos definido en nuestra base de datos musical.

9. Referencias

<https://www.python.org>

https://www.tensorflow.org/versions/r0.10/get_started/os_setup

<https://es.wikipedia.org/wiki/Pip>

<http://pandas.pydata.org>

[https://en.wikipedia.org/wiki/Pandas_\(software\)](https://en.wikipedia.org/wiki/Pandas_(software))

<http://msgpack.org>

<https://github.com/tqdm/tqdm>

<https://github.com>

<https://bitbucket.org>