Deep Learning: Neural Networks for Generating Music

Manuel Santiago de Toro

02 de julio 2018

Resumen– Durante los últimos años se está debatiendo si la inteligencia artificial es capaz de generar arte como si de una persona se tratase. El objetivo de este proyecto es crear música de manera artificial sin que se pueda apreciar si ha sido creada por una máquina o por un artista. Por consiguiente, se realizan diferentes experimentos para determinar si es mejor entrenar con un conjunto de datos especializados de entrenamiento o con un conjunto más amplio sin especializar. Además, se quiere observar cómo influye el batch size en este entrenamiento. Como consecuencia, para ello, la base se parte des del proyecto Folk-rnn [1] de IraKorshunova. Asimismo, la generación de música folk irlandesa se realiza mediante una red neuronal de tipo recurrente (RNN). Ésta posee bloques de larga memoria a corto plazo (Long Short-Term Memory LSTM) y está implementado en python con la librería Theano. Por lo que respecta al formato que se utiliza para entrenar la red y crear música es ABC [2] y se tratan las notas musicales como caracteres.

Palabras clave– RNN, LSTM, música, folk, ABC, generación, deep learning, redes neuronales, arte, Theano.

Abstract– During the last years, it is being debated whether an artificial intelligence is capable of generating art as if it was a person. The objective of this project is to create music in an artificial way, without have been able to appreciate if it has been created by an artist or a machine. In addition, different experiments are carried out to determine if training with a specialized data set of training is better than with a larger set without specializing and how batch size influences training. This is part of the project Folk-rnn [1] IraKorshunova. Likewise, the generation of Irish folk music is carried out through a recurrent type of neural network (RNN). It has blocks of long term memory (Long Short-Term Memory LSTM) and is implemented in python with the Theano library. Regarding the format that is used to train the network and create music is ABC [2] and treat the musical notes as characters.

Keywords– RNN, LSTM, folk, ABC, deep learning, theano, IA, creative

1 Introducción

Durante los últimos años la inteligencia artificial está creando bastante controversia por el hecho de si ésta es capaz de generar arte como si se tratara de una persona. En la actualidad, frente al auge en el uso de las redes neuronales, se encuentran diferentes proyectos que intentan generar música con toques originales y artísticos. Así pues, el principal objetivo de éstos es crear música que no permita diferenciar si ha sido creada por una máquina o por un gran artista.

Para realizar el trabajo me he centrado en el proyecto de Ira Korshunova folk-rnn [1] en el cual crea una red neuronal que es entrenada y que genera música folk irlandesa con buenos resultados. Se basa en la red char-rnn de KARPATHY [14] que es una red de tipo recursiva para generación de texto automático. Cabe destacar que se pretenden realizar mejoras en la generación de música, analizar cómo influyen los datos de entrada en esta red, evaluar si la música generada es semejante a la creada por un músico y, por último, qué errores son los más comunes que se producen en las canciones creadas. El formato de la música empleado en folk-rnn es ABC. Del mismo modo, éste representa las notas musicales a través de caracteres por lo que con una canción se conseguiría una cadena de caracteres. Por esta

• E-mail de contacto: manuel.santiago@e-campus.uab.cat
• Mención realitzada: Computación
• Tutor: Fernando Luis Vilarino Freire (Departamento de ciencia de la computación)
• Curso 2017/18

razón, se asemeja a la red char-rnn.
El presente trabajo está distribuido en 8 secciones. En la primera se encuentra la introducción. En la segunda sección se describen los objetivos a los que se pretende llegar. Y, en el tercer apartado, se efectúa un repaso al estado del arte, dándole una pincelada a las redes neuronales recurrentes, y se expone la historia de la generación de la música automática y se describe la folk-rnn. A continuación, en la cuarta parte muestro la metodología empleada para llevar a cabo este trabajo: cómo preparar el data set, entrenar la red y generar música. En el quinto apartado están los resultados obtenidos, los cuales intentan responder a las preguntas formuladas en los objetivos. Así pues, en la penúltima sección se encuentran las conclusiones a las que he llegado y, para acabar, están los anexos.

2 Objetivos

A continuación se expondrán los objetivos de este proyecto:

1. Creación de música artificial
2. Estudio de una red neuronal de tipo recursiva
3. Implementar la red y analizar los resultados
4. Analizar resultados al entrenar con distintos data sets
5. Dar una respuesta a las siguientes cuestiones:
   - 5.1 ¿Influye la tonalidad con la que se entrena la red en el resultado de las canciones creadas?
   - 5.2 ¿Son diferentes las canciones originales respecto a las artificiales?
   - 5.3 ¿Qué errores cometen las redes neuronales en la creación de música?
   - 5.4 ¿Influye el tamaño del batch en el entrenamiento?

3 Estado del Arte

3.1 Redes neuronales recurrentes

Las redes neuronales recurrentes son sistemas cuya salida depende de los datos de entrada actuales y también de los datos que ya han sido tratados en el pasado. En éstas se incorpora un estado variante en el tiempo y se caracterizan por tener capacidad de memoria. Asimismo, este tipo de redes se utilizan sobre todo para tareas que involucran datos secuenciales como por ejemplo: la creación de texto, el reconocimiento de voz, la creación de vídeo y música, la descripción de imágenes, etc. Cabe añadir que las redes recurrentes son más poderosas computacionalmente en comparación con las redes feedforward (por ejemplo MLP o CNN). En éstas últimas, cualquier problema que resuelvan, puede ser transcrito para ser solventado por una RNN poniendo los datos en forma secuencial)[6].

Por lo tanto, las RNN además de sus pesos, tienen un estado (hidden state). La misma función y el mismo conjunto de parámetros son usados en cada paso.

3.1.1 Tipos de redes recurrentes

- One to one: Red no recurrente, de entrada fija a salida fija.

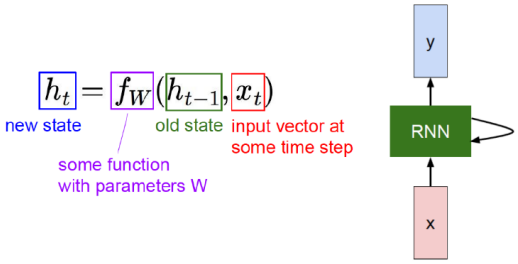


Fig. 1: Función RNN

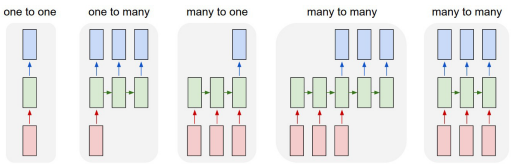


Fig. 2: Tipos de RNN de izquierda a derecha: one to one, one to many, many to one, many to many, many to many sincronizada[7]

- One to many: Salida secuencial.
- Many to one: Entrada secuencial.
- Many to many: Entradas y salidas secuenciales.
- Many to many: Entradas y salidas secuenciales sincronizadas.

3.1.2 Conexiones recurrentes

La conexiones entre las neuronas puden ser de diferente tipo:

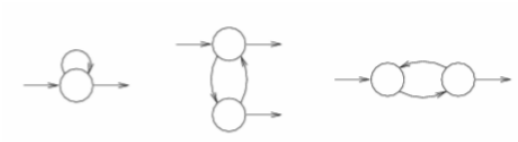


Fig. 3: Conexiones[8], con sigo misma, entre neuronas de la misma capa y conexión con neuronas de la capa anterior

- De una neurona con ella misma.
- Entre neuronas de una misma capa.
- Entre neuronas de una capa a una capa anterior.

3.2 Memorizar estados

El principal problema que presentan las RNN convencionales en su entrenamiento es que los gradientes retro propagados tienden a crecer o desvanecerse con el tiempo (Vanishing gradient-Explonding gradient). Esto es debido a que el gradiente depende tanto de los errores presentes como de los pasados.
Por este motivo, se han desarrollado diferentes arquitecturas y métodos de aprendizaje que evitan estos problemas como son las Long Short-Term Memory (LSTM) o la Clockwork RNN.
La idea central detrás de las LSTM es una celda de memoria que puede mantener su estado en el tiempo y, además, compuertas no lineales que permiten regular el flujo de la

información por dentro y por fuera de la celda.

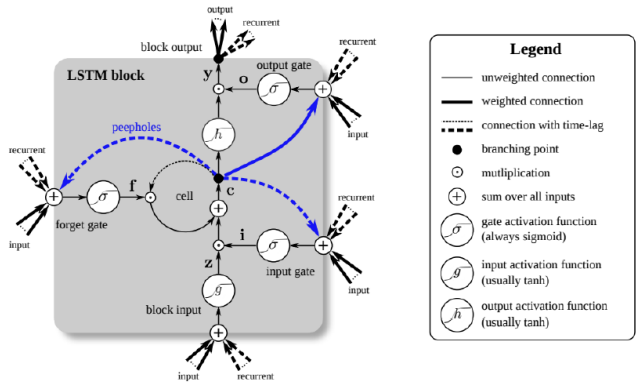


Fig. 4: Long Short-Term Memory (LSTM)[6]

En cuanto a las Clockwork RNN, la capa oculta es separada en módulos y cada módulo tiene su propia frecuencia reloj a la cual computa sus operaciones. No obstante, aunque esto parece aumentar la complejidad del sistema, en realidad reduce el número de parámetros a entrenar y acelera la evaluación de red.

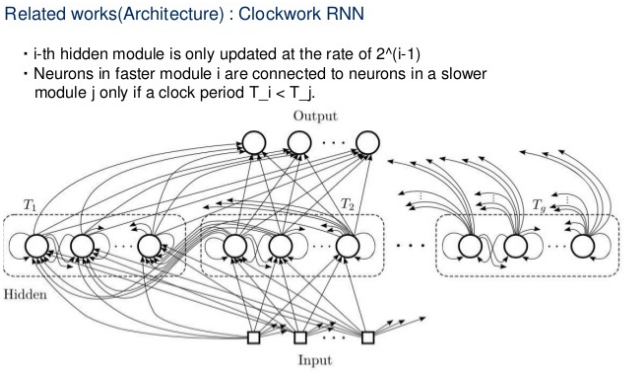


Fig. 5: Clockwork

Según el artículo de JanKoutník[et al.] A CLOCKWORK RNN [9] la diferencia en los resultados finales entre una LSTM-RNN y una CW-RNN es mínima. Sin embargo, la ventaja de las CW-RNN es que precisa entrenar con menos datos para obtener resultados similares. Por ello, necesitará menos tiempo para entrenar.[10]

3.3 Historia de generación de música automática

Durante la historia, todas las culturas han creado diferentes tipos y estilos de música. Muchas de éstas han buscado música nueva y original mediante sistemas aleatorios de generación. Así pues, en las últimas décadas gracias a diferentes algoritmos y herramientas como son las redes neuronales, se están consiguiendo grandes resultados. En la siguiente Tabla 1 podemos ver un breve cronograma de la generación de música aleatoria.

TABLA 1: BREVE HISTORIA DE LA GENERACION AUTOMATICA DE MUSICA[11]

Año	
1100 a.c	**Windchimes**: en China, sonido aleatorio producido por el viento) **Suikinkutsu**: en Japón, sonido aleatorio producido por corrientes de agua)
1700	**Musikalisches Würfelspiel**: se hacía uso de dos dados para generar música de forma aleatoria, los dados seleccionan al azar breves pasajes de música.
1900	**Cadenas de Márkow**: utiliza probabilidades para generar notas dependiendo de la nota anterior producida.
1981	David Cope **combina cadenas de Márkow con otras técnicas** inspiradas en el trabajo de Iannis Xenakis.
1989	**Primer intento de generar música con RNNs** por Peter M, Todd Micheal C. Mozer. Estuvo limitada por la corta memoria y la falta de coherencia.
2002	**RNN-LSTM** Doug Eck utiliza por primera vez las celdas LSTM para generación automática de música.
2015	**Char-rnn + notación simbólica de caracteres**: se emplea una nueva red capaz de generar texto y se utiliza para generar música.
2016	**WaveNet**[12] es publicada por DeepMind: arquitectura que puede construir abstracciones de nivel superior de audio muestra por muestra. Se basa en CNN.
2017	**NSynth** El equipo de magenta con colaboración de Google Creative Lab presenta un modelo para analizar y generar música monoinstrumental basada en WaveNet.
2017	**SampleRNN** publicada por Yoshua Bengio genera audio usando RNN en estructuras jerárquicas
2017	**Fast-Wavenet**: WaveNet optimizada usa audios de 16kHz and 8bit. Dado que necesitan un largo periodo de entrenamiento es necesario hacer todas las optimizaciones que sean posibles.

3.4 Folk-rnn

El objetivo del proyecto folk-rnn es la creación de canciones artificiales con una gran calidad utilizando redes neuronales. Como consecuencia, se usa para el entrenamiento música de estilo folk irlandesa ya que este tipo de música tiene una gran base de datos en el formato que se requiere en TheSession.org.

El formato que se emplea es el ABC ya que éste representa las notas musicales con caracteres y, de este modo, permite tratar la música como cadenas de caracteres. Así pues, se utilizan muchos conceptos del proyecto de KARPATHY, char-rnn para crear la red.
La red que usa es de tipo recurrente con LSTM para así memorizar los estados anteriores que utiliza para los resultados que se producen.
La arquitectura que dispone es la siguiente:

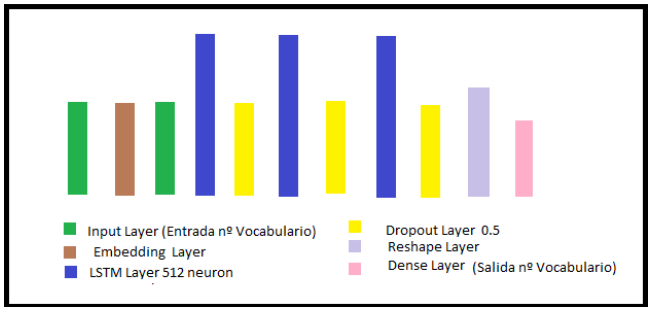


Fig. 6: Arquitectura de la red folk-rnn

Cabe decir que para las capas LSTM aplica la función de activación tangente hiperbólica y, para la capa de salida Dense, la función SoftMax ya que se quiere obtener probabilidades y ésta es muy buena para este propósito. En la Folk-rnn la función de pérdida que se utiliza es la entropia cruzada. Además, existen dos versiones dependiendo del data set que se utiliza para el entrenamiento. El data set se construye a partir de un conjunto de 23,958 canciones obtenidas en TheSession.org[15]

- V1: elimina del data set original los siguientes campos de la cabecera X:,Z:,S:, y R:.
- V2: como la V1 pero también elimina los campos T: y L:, quita melodías con diferentes Claves y múltiples voces, suprime adornos y *gracenotes* y transpone todas las melodías para tener la tónica C (dando así cuatro modos: major, mixolydian, dorian y menor).

```
T: Bornity Horse
M: 4/4
L: 1/8
K: Dmaj
|:FG A2 BGBd|AFGE DEdB|AF  A2  BGBd|(3cBA cd eAFG|
FA  B2 ABde|fedB AF F2|BG G2 ABdf|1 afea fdd2:|2 afea ~d3z||
|:d3e fd d2|Bd d2 Adfd|effe dff2|afeg fd d2 |
defd ed B2|ABde faaf|a2 fd Bd d2|AFde fdeg||
```

Fig. 7: Ejemplo de salida con data set V1 folk-rnn

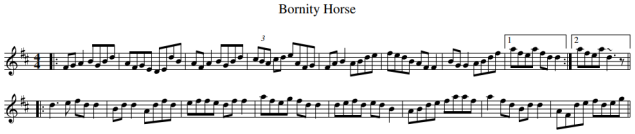


Fig. 8: Pentagrama de la saida con data set V1 folk-rnn

3.5 Cómo genera música la folk-rnn

A la hora de entrenar la folk-rnn se genera una red con unos determinados pesos que permiten saber la probabilidad de

```
M:4/4
K:Cmaj
|: G 2 E > C G > C E > C | G 2 E > G c > G E > C | D 2 D > F (3 D E D [ B, C ] > E |
C 2 (3 E C C B > F D > A |G 2 E > C E > C E > C | G 2 E 2 G > C E > C | D 2 d 2 G > B
d > B |1 c 4 c 2 (3 G A B :| |2 c 2 B > A C 3 G /2 A /2|: B > c d > e f > d B > c | d
> e d > e c > A (3 G A B | c 2 e > c c 4 | e > a (3 e e e c 2 (3 G A B |c 2 c > e f >
e d > c | _B 2 B 2 A < B d < c | B > G F > D D > _B B < d |1 c 2 B 2 c > A G < B :|
|2 c 2 B 2 c 2 c 2 |
```

Fig. 9: Ejemplo de salida con data set V2 folk-rnn

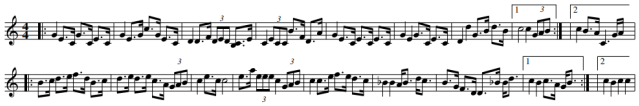


Fig. 10: Pentagrama de la saida con data set V2 folk-rnn

cada carácter que se tiene para generar el próximo, ejemplo 11

Vocabulario {A, B, C, D}	
Secuencia generada ”	Generado por la red entrenada A=0.50 B=0.25 C=0.25 D=0
Se escoge un carácter Aleatorio teniendo en cuenta la probabilidad	
‘A’	Probabilidad dada la secuencia ‘A’ generada por la red entrenada A=0.15 B=0.40 C=0.30 D=0.15
Se escoge un carácter aleatorio teniendo en cuenta la probabilidad, no siempre se escoge el carácter con mayor probabilidad	
‘AC’	Probabilidad dada la secuencia ‘AC’ generada por la red entrenada A=0.05 B=0.10 C=0.05 D=0.80
‘ACD’	...

Fig. 11: Ejemplo de los pasos que sigue folk-rnn para generar canciones en ABC

4 METODOLOGÍA

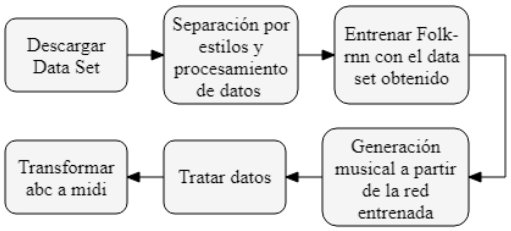


Fig. 12: Diagrama de flujo basico para la generación musical

4.1 Data set y proceso de datos Fig.13

Para preparar el data set lo primero que se ha realizado ha sido buscar música en el formato ABC (que es con el que trabajaré a través de Internet). Tras una búsqueda y leer detenidamente la información de Ira Korshunova[1] que tiene en su Git, encontré una gran base de datos de canciones https://thesession.org/[3].

4.1.1 Descarga de archivos

El primer paso es la descarga de ficheros que contienen las canciones en formato ABC de la web[3]. Para acceder a

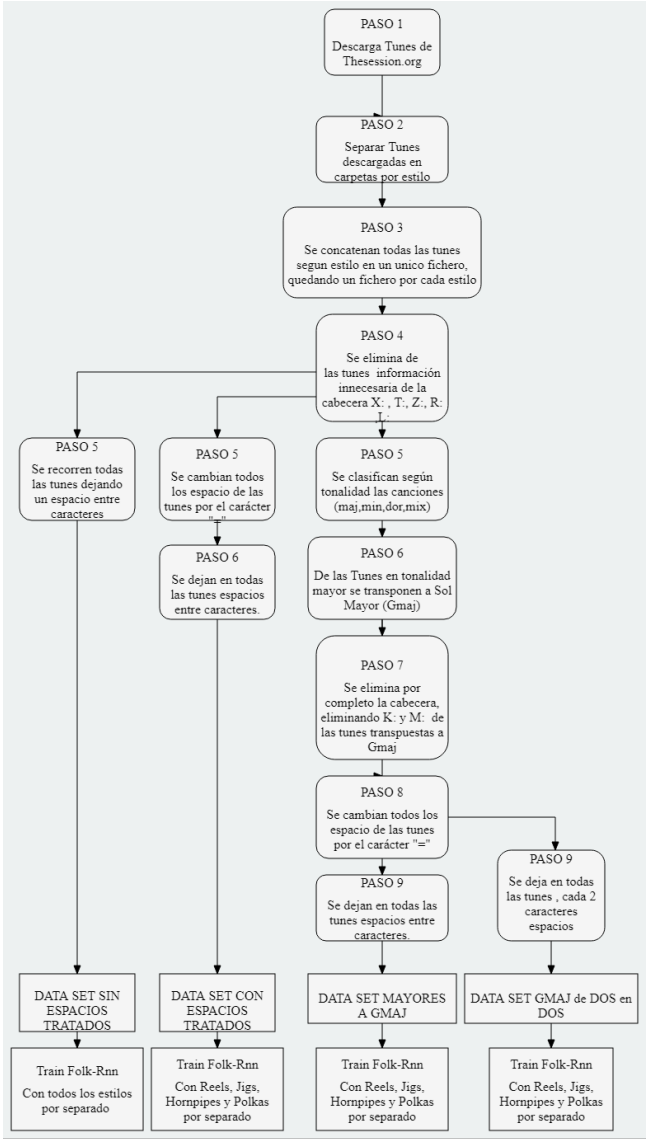


Fig. 13: Metodología, preparación de los diferentes data sets

cada una de las canciones a través de url se debe de cambiar **N** *https://thesession.org/tunes/**N**/ABC*. Asimismo, para ello, creé un scrip en Python que consiste en ir incrementando N y descargando el fichero hasta que haya una secuencia de 20 N seguidas que no contengan ningún fichero para descargar. Esto indica que se ha llegado al final de la secuencia de ficheros válida para descargar. Es remarcable añadir que el número 20 es para asegurar que se ha llegado al final puesto que pueden haber algunos números que no contengan fichero, por ejemplo *https://thesession.org/tunes/**100**/ABC*. De esta manera se consigue un data set de mas de 16000 canciones.

4.1.2 Clasificación según el estilo musical

Una vez descargados los ficheros procedemos a la clasificación según el estilo musical. Cada archivo ABC contiene el estilo al que pertenece en la cabecera (R:"estilo"), por lo tanto, se debe leer cada fichero y guardarlo en la carpeta que corresponda a su estilo. Además, aproveché y lo guardé con el nombre del título en la carpeta correspondiente. Después de ejecutar el scrip tengo todas las canciones clasificadas en carpetas por estilos con el nombre de archivo y el nombre del título (cabecera T:"título")

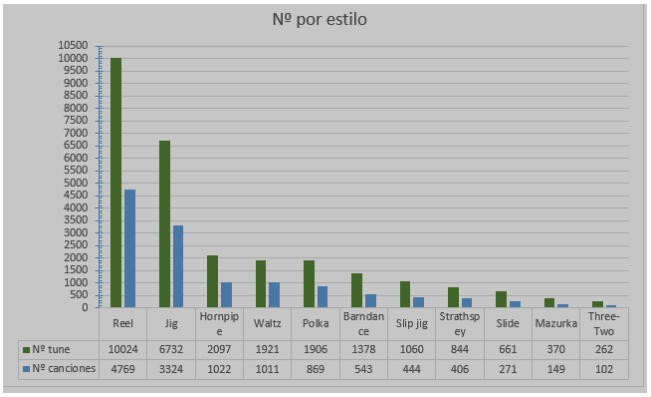


Fig. 14: Numero de muestras de cada estilo

No obstante, para algunos estilos como Strathspey, Slide, Mazurka, Three-two no hay muestras suficientes para entrenar la red y obtener buenos resultados. Por lo tanto, nos centraremos sobre todo en los estilos con más muestras como son Reel, Jig, hornpipe y polka.

4.1.3 Juntar todos los archivos del mismo estilo

El siguiente paso es juntar todos los archivos del mismo estilo en uno único para de este modo poder entrenar con él. Con esto quedaría un archivo por cada estilo con todas las canciones dentro de él.

4.1.4 Quitar datos de cabecera

Hay información poco útil en la cabecera. Por consiguiente, procedo a eliminar todos los datos que no me ayudan a obtener mejores resultados en la calidad musical como T:título, Z:autor, R:estilo, S:fuente de descarga, etc. De esta forma sólo me quedo con la tonalidad y la clave (K: y M:) para los data set sin espacios tratados y data set con espacios tratados. Se puede observar que para los data sets Mayores a Cmaj y data set Cmaj de 2 en 2 se elimina por completo la cabecera porque según el estilo tienen una clave en concreto y la tonalidad es Cmaj. Sin embargo, luego se le incluye otra vez una vez creada la música.

Estilos	M:
Barndance	4/4
Hornpipe	4/4
Jig	6/8
Mazurka	1/8
Polka	2/4
Reel	4/4

TABLA 2: CLAVE SEGÚN ESTILOS

4.1.5 Dejar espacio entre caracteres. Data Set espacios sin tratar

Para solucionar el problema del gran vocabulario tratamos los caracteres de forma individual. Para ello, dejo entre carácter y carácter espacio en blanco ya que el código de train coge como vocabulario los datos del data set que están separados por espacios.

Por el contrario, si se tratara de texto en vez de entrenar con palabras, se entrenaría con letras reduciendo el número (palabras = diccionario, letras = abecedario). Así pues, vemos

'D'	'H'	'L'	'P'
'0'	'4'	'8'	'<'
'+'	'/'	'3'	'7'
'm'	'q'	')'	'u'

TABLA 3: VOCABULARIO DEJANDO ESPACIOS ENTRE CARACTERES (REEL(V = 158)

reducido el número del vocabulario para reels de 89274 a 158. Por este motivo, ahora sí que se puede proceder a entrenar la red con este data set.

4.1.6 Tratar el carácter espacio como otro más. Data Set con espacios tratados

Sin embargo, los resultados data set sin espacios tratados no son los esperados. La razón es porque no se está tratando el carácter espacio como si de uno se tratase. Es por ello que cuando se separan los caracteres por espacios éste se pierde. Además, la red crea música con notas separadas dando un sonido muy monótono y de poca calidad. Por tanto, para solucionar este problema cojo el data set sin espacios. Lo primero que se hace es cambiar los espacios por signo = y luego se procede a dejar espacios entre caracteres. Y cabe decir que posteriormente trataremos el signo = como espacios.

4.1.7 Especialización estilo tonalidad maj,min,dor y mix. Data Set mayores a Gmaj

Para especializar más la red e intentar obtener mejores resultados, se separan todas las canciones de un mismo estilo según la tonalidad maj, min, mix, dor. Una vez ejecutado este paso, se realiza una transposición de tonalidad dejando todas las canciones mayores en Sol mayor (Gmaj).
A continuación se procede a eliminar por completo la cabecera ya que la tonalidad (K:) de todas estas canciones será GMaj y, la clave (M:), viene determinada por el estilo y este lo sabemos.
En consecuencia, con este data set la red se entrenará con un estilo y tonalidad determinados. Se puede exponer que se realiza con la tonalidad maj porque es la más significativa Tabla 4 y 5

TABLA 4: Nº DE CANCIONES Y % POR TONALIDAD HORNPIPE-JIG

HORNPIPE		JIG	
Maj = 3477	89.2%	Maj = 4525	59.1%
Min = 198	5.08%	Min = 1138	14.8%
Dor = 157	4.03%	Dor = 806	10.5%
Mix = 62	1.59%	Mix = 1182	15.4%

TABLA 5: Nº DE CANCIONES Y % POR TONALIDAD REEL-POLKA

REEL		POLKA	
Maj = 6233	48.0%	Maj = 1507	76.4%
Min = 3291	25.08%	Min = 208	10.5%
Dor = 1745	13.4%	Dor = 187	9.49%
Mix = 1716	13.2%	Mix = 68	3.45%

4.1.8 Tratar caracteres de 2 en 2. Data Set Gmaj de 2 en 2

Se prepara el data set partiendo desde el data set Mayores a Gmaj 4.1.7 separando de dos en dos los caracteres. Así, la red cogerá como vocabulario palabras de dos caracteres en vez de cogerlos de 1 en 1 para entrenar.

TABLA 6: TRANSPOSICIÓN DE TONALIDAD MAYOR A GMAJ POR NUMERO DE SEMITONOS

Tonalidad	Nº de Semiton
CMaj	7
DMaj'	5
EMaj'	3
FMaj'	2
GMaj'	0
BMaj'	-4

4.2 Entrenamiento de la red

Una vez tratados todos los datos y preparado todo el conjunto de data set que utilizaré, procedo a entrenar la red. La arquitectura de la red es la misma que la del proyecto folk-rnn y, dado que daba buenos resultados, no he visto apropiado cambiarla. El número de epoch utilizados varían entre 75 y 300, al principio entrenaba con 300 pero al ver que no era necesario realizar tantos ya que no mejoraba el resultado incluso empeoraban por el overfiting, decidí bajarlo de manera consecutiva. El learning rate que he utilizado ha sido de 0.003. La muestra que se escoge como validación es del 10% del data set de entreno y se escoge de manera aleatoria. Utilizo 56 de tamaño del batch porque con los estilos con más muestras como los Reels, era el máximo que permitía la tarjeta gráfica con la que se entrenaba (ver hw de anexoA.7) ya que se necesita una gran capacidad de memoria. Además incorporo en cada epoch una verificación del loss y, si mejora, guarda los resultados. Al final se obtiene una red con los pesos del mejor loss conseguido y otra con los pesos del último epoch.
También incluyo la realización gráfica de la evolución de los valores de la función loss durante el train.

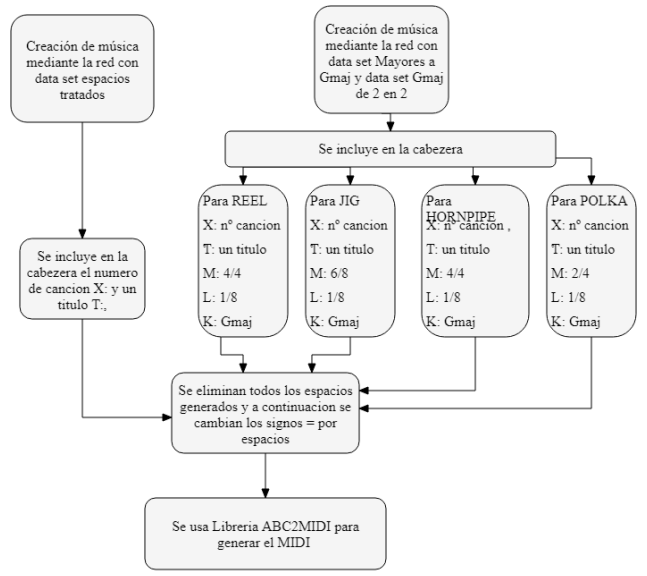


Fig. 15: Pasos para generar MIDI, ampliado en A.9

5 Resultados

Los resultados son preliminares que no buscan un análisis en profundidad en significación estadística p values. Para obtener éstos se ha empleado la red folk-rnn, sección 3.4. Asimismo, se entrena con los data set ya mencionados anteriormente.

- Sin tratar espacio (con todos los estilos disponibles fig.14)
- Tratando espacios (con estilos Reel, Jigs, Hornpipe y Polkas)
- Mayores a Gmaj (con estilos Reel, Jigs, Hornpipe y Polkas)
- Gmaj de 2 en 2 (con estilos Reel, Jigs, Hornpipe y Polkas)

- Reel Con espacios tratados - 151 voc - 10024 tune
- Reel Mayores a Gmaj - 103 voc - 6233 tune
- Reel Gmaj 2 en 2 - 2049 vocabulario - 6233 tune

- Jig Con espacios tratados - 146 voc - 10024 tune
- Jig Mayores a Gmaj - 95 voc 4525 tune
- Jig Gmaj 2 en 2 - 1826 voc - 4525 tune

Para jigs y reels en todas las versiones se obtiene el mejor resultado antes del epoch 100. Por lo tanto, no sería necesario entrenar la red más de 100 epoch ya que, una vez alcanzado el mínimo, poco a poco va aumentando el loss debido a que se produce overfiting. En cambio, para la versión con data set Gmaj 2 en 2 el overfiting es mayor, posiblemente porque el tamaño de caracteres de entrada (vocabulario) es más considerable en comparación con las demás versiones. Podemos percibir que los resultados de entrenamiento son muy similares, independientemente del estilo que se entrene. No obstante, las mayores variaciones son debido al número del vocabulario.

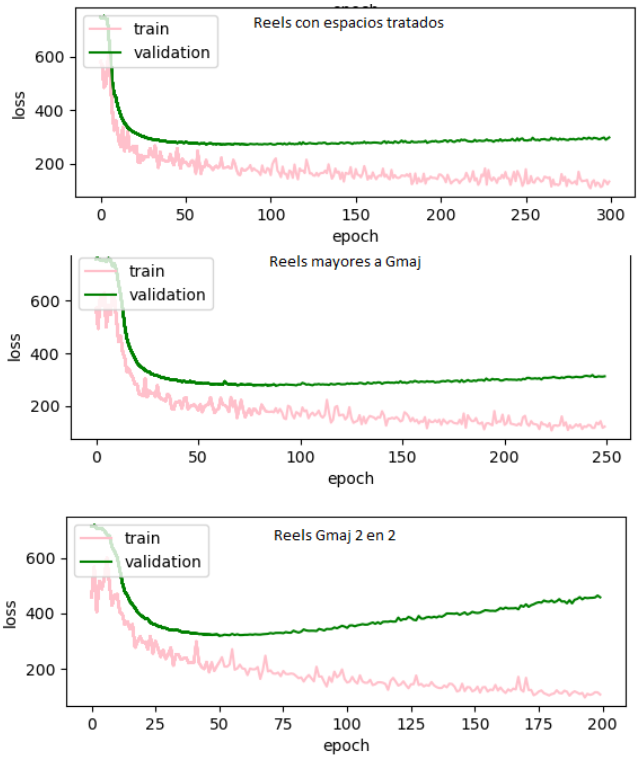


Fig. 16: evolución loss-epoch en Reels

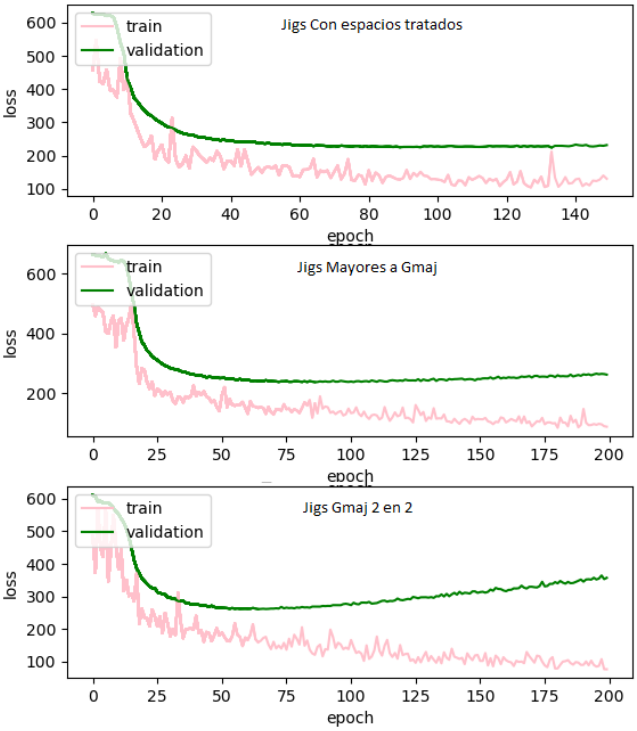


Fig. 17: evolución loss-epoch en Jigs

5.1 ¿Cómo influye el tamaño del Batch a la hora de entrenar?

Por limitaciones de hardware, muchas veces no se puede utilizar el batch size desado puesto que la memoria de la tarjeta gráfica es limitada. Por esta razón hago un pequeño estudio sobre cómo influye. Entre el tamaño de 1-6 en los resultados del loss se producen picos irregulares. Al coger los pesos que resultan con el mínimo loss que se ha pro-

ducido durante el entrenamiento, observo que esto no me influye (sólo miré el mínimo al que se llega). Cabe añadir que a partir del tamaño 18 la mejora en el mínimo loss es insignificativa.
Del tamaño 8 al 56 lo mas diferenciable es el tiempo que se necesita de entrenamiento para conseguir los epoch deseados.
También se analiza que a tamaño mayor del batch se necesitan más epoch para llegar al mínimo. Como consecuencia puedo decir que a partir de un tamaño del batch, en nuestro caso 8, la función del loss mejora muy poco y el tiempo es lo más significante. Por lo tanto, el batch size debería estar entre 8 y 56 (o máximo que permitiera el hardware).

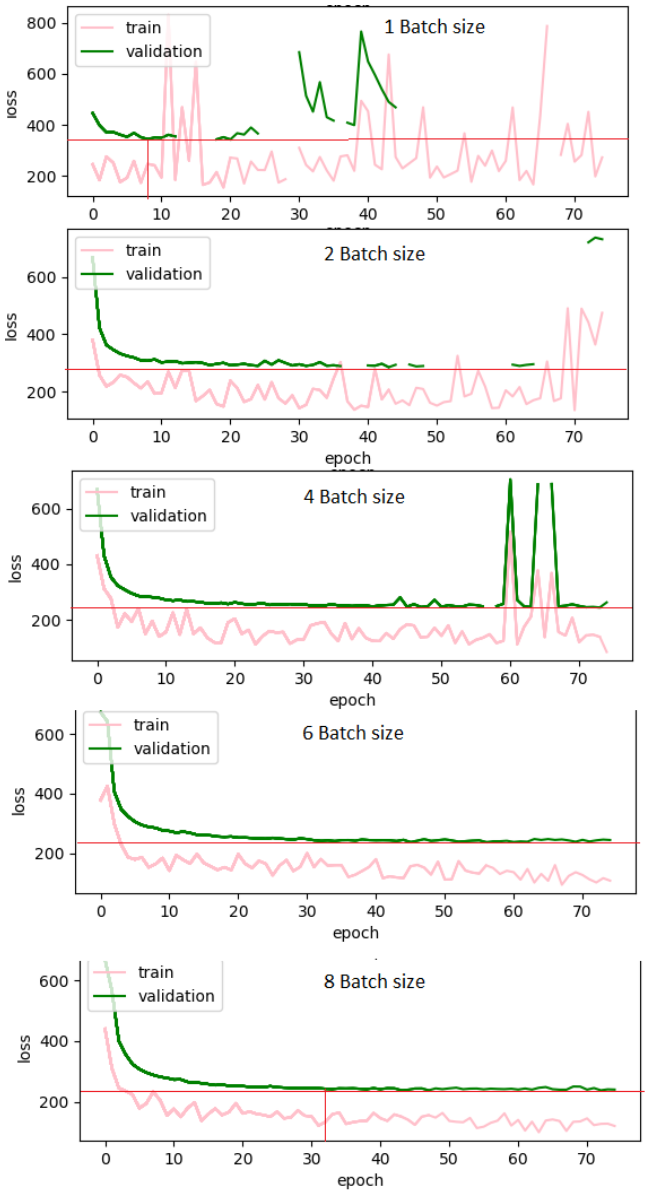


Fig. 18: Batch size

5.2 ¿Influye la tonalidad con la que se entrena la red en el resultado de las canciones creadas?

Para resolver esta pregunta he realizado la siguiente prueba: con la red del proyecto folk-rnn que utiliza todo el data set del TheSession.org sin separar por estilo ni cambiar tonalidad a Gmaj, creo 3025 canciones. Seguidamente escojo los Jigs porque de la cabecera sólo conserva K: Tonalidad y M: clave y sólo se puede distinguir el estilo por la clave. Percibimos que la clave del jigs es 6/8 y no es compartida con más estilos, igual que el de la polka que es 2/4. Es destacable remarcar que con reels no se podría ya que la clave es 4/4 y lo comparte con muchos otros estilos como barndance, hornpipe, etc. y no sabríamos si se trata de un reel real o de otro estilo. De esta forma obtengo 1448 Jigs de los cuales 998 son mayores. Así pues, los mayores los transponemos a Gmaj para poderlo comparar con el siguiente conjunto.
Por otra parte, creamos 500 canciones Jigs con la red entrenada con el data set de Jig- mayores a Gmaj. Estas canciones creadas ya están en la tonalidad Gmaj. Una vez tenemos los dos conjuntos de canciones, procedemos al análisis.

TABLA 7: TIEMPO DE ENTRENO Y EPOCH DONDE SE ALCANZA EL MÍNIMO EN LA FUNCIÓN LOSS SEGÚN EL TAMAÑO DEL BATCH

NºBatch	Time Train	Min Loss 75 epoch:
1	540 min	350
2	520 min	300
4	511 min	260
6	492 min	252
8	480 min	250
18	330 min	245
56	92 min	243

5.2.1 Cuantitativo

Para el análisis cuantitativo me centro en el estudio de las frecuencias de los caracteres de las canciones creadas por los dos conjuntos.
Como se puede observar en la Tabla 8, las frecuencias de los caracteres en la aparición de las canciones creadas es distinta. Hay una notable diferencia de entrenar con todo el data set como lo hace folk-rnn a entrenar con sólo un estilo y una tonalidad en concreto.
Los jigs generados con todo el data set cogen particularidades de otros estilos y los combina, no diferencia entre estilos. Por lo que si se desea conseguir un estilo y tonalidad en concreto es mejor entrenar la red con un data set especializado.

TABLA 8: FRECUENCIA EN LA CANCIONES CREADAS POR FOLK-RNN VS MAYORES A GMAJ

Data set completo folk-rnn		**Data set Mayores a Gmaj**	
Carácter	Porcentaje	Carácter	Porcentaje
d	13,46%	d	12,22 %
g	9,24%	B	11,24 %
B	8,97%	G	7,46 %
e	6,98%	g	7,00 %
G	6,80%	f	6,58 %
c	6,07%	e	5,94 %
A	5,69%	A	5,86 %
b	5,69%	c	5,67 %
a	5,66%	b	3,59 %
f	3,35%	a	3,34 %

5.2.2 Cualitativo

Para analizar de forma cualitativa, escogí de forma aleatoria 3 canciones Jigs pasadas a Gmaj (generadas con todo el data set) y otras 3 canciones aleatorias con el data set Jigs Mayores a Gmaj. Además, hice una encuesta [19]. El número de participantes que la han realizado ha sido 20.
La encuesta consiste en evaluar las canciones escogidas al azar y la persona valora en función de si le ha gustado más o menos. Para cada una de ellas se valora entre una puntuación del 0 al 5 siendo 0 la peor nota y 5 la mejor posible.

Los resultados obtenidos son los siguientes:

TABLA 9: VALORACION MUSICAL FOLK-RNN VS MAYORES A GMAJ

Cancion	Puntuacion Media	Pertenece
1	55	mayores a Gmaj
2	51	folk-rnn
3	69	folk-rnn
4	76	mayores a Gmaj
5	65	mayores a Gmaj
6	64	folk-rnn

Como se observa en la Tabla 9 las canciones creadas con folk-rnn tienen una puntuación de 184 puntos y las canciones creadas por el data set mayores a Gmaj obtienen una puntuación de 196. Por lo tanto, reciben una mejor valoración las canciones creadas con el data set Jigs Mayores a Gmaj habiendo una diferencia de 12 puntos. Como consecuencia, no hay una clara diferencia para poder afirmar que una es mejor que la otra, por lo que necesitaríamos un estudio estadístico más completo con más muestras y más participantes (a ser posible expertos).

5.3 ¿Son diferentes las canciones originales respecto a las artificiales?

Analizo si las canciones creadas por una red neuronal son similares o por el contrario fácilmente diferenciables a canciones creadas por músicos. Para ello realizaremos un estudio comparativo de las canciones que se usan para entrenar la red que son creadas por seres humanos(canciones reales) con las generadas por la red(canciones artificiales). Por consiguiente, se empleará el Data Set de Reels con espacios tratados y, también, se comparará el data set de entrada con 1000 canciones creadas después de entrenar la red con este data set.

5.3.1 Cuantitativo

TABLA 10: FRECUENCIA DE CANCIONES REALES Y CANCIONES ARTIFICIALES CREADAS POR LA RED NEURONAL CON EL DATA SET CON ESPACIOS TRATADOS

Data set con espacio Tratado canciones originales		**Data set con espacio tratado canciones generadas**	
Carácter	Porcentaje	Carácter	Porcentaje
A	7.25%	A	7.11%
d	6.72%	d	6.94%
B	6.64%	B	5.90%
e	5.53%	e	5.79%
2	4.92%	f	5.15%
c	4.46%	2	4.91%
G	4.43%	G	4.52%
f	4.02%	g	4.21%
g	3.36%	c	3.80%

Hay que tener en cuenta que los caracteres que aparecen menos del 0.013% en el data set de entrenamiento no aparecen luego en las canciones generadas.
Los seis primeros caracteres de la tabla son idénticos y el porcentaje de aparición varía poco. Sin embargo, algunos de los siguientes caracteres varían un poco más pero otros son prácticamente iguales (G). La variación de frecuencia entre los caracteres de canciones reels creadas por personas y creadas por la red con el data set reels con espacios tratados es mínima. Es destacable comentar que la red neuronal mantiene una gran parte de las frecuencias de las canciones originales.

5.3.2 Cualitativo

Para analizar de forma cualitativa escogí de forma aleatoria 3 canciones Reels Originales y otras 3 canciones aleatorias creadas con el data set Jigs espacios tratados e hice una encuesta [19] donde participaron 20 personas.
La encuesta consiste en detectar si las canciones están hechas por personas (real) o por la red neuronal (artificial). Los resultados los mostraré con una matriz de confusión:

		Valor Predicho	
Valor Real		Humano	Artificial
	Humano	26	31
	Artificial	34	29

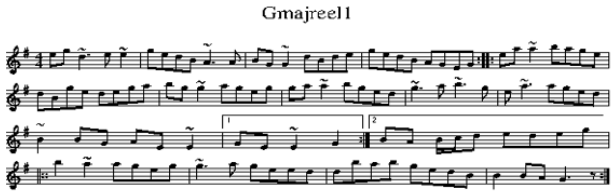
Fig. 19: Matriz de confusion de los resultados de la encuesta realizada en Humano contra Artificial

Mean error = (FP + FN) / N = (31+34)/120 = **0.542**
Preccision = TP / (TP+FP) = 26/(26+31) = **0.456**
Accuracy = ((TP + TN) / N)= (26+29)/120 =**0.458**

Como se puede observar, los encuestados no notan una diferencia clara entre las canciones artificiales y las humanas. Esto puede ser debido a que la mayoría de encuestados no tienen grandes conocimientos de música. Así pues, se tendría que analizar con más muestras y expertos en música. No obstante, la primera impresión es que la red es capaz de generar música muy próxima a la música que crearía un artista.

5.4 ¿Qué errores cometen las redes neuronales?

Para esta pregunta utilizaremos las canciones obtenidas con la red del data set de Reels Gmaj de 2 en 2 ya que se han obtenido peores resultados y es mas fácil ver los fallos.



T:Gmajreel1
eg ~d3 e~e2 | gedB ~A3 A | BG ~G2 dBde | gedB AGEG :| |: ea ~a2 bage | dBge dega | bg ~g2 ageg | ga~a2 bged |~g3 a ~b3 g | e~a3 aged | ~B2 BG AE ~E2 |1 GE ~E2 G2 :|2 BA B/c/d edeg |||:: b2 ~a2 ageg | ~g3 a geed | dbab gedB | B2 BA G3 z :|

T:Gmajreel9
|:Acde dBGE|D2DE GEDE|DGBG Adge|1dBAe G3E:|2dedB AGAB||||:gedB AGBd|gdBd gdBd|eaae afg2|abge dBAB|DGEG DEGA:|

T:Gmajreel14
|:a-- a2 eddb|g}fa fe g|fdf-e-eff|{df}-efef|gagf BGGe|Ggfe Ad^cA|fage dfaf|defd g4:||:kaA]1Ffed eGga|faef dgfe|f/2f..ff/e]/2Bd fg:|

T:Gmajreel18
GA|:B2AB G2ge|dBBA G2GE|EDEC EEDF|B2BA GABd|gfge dBAG|EABB AGEG|DGAB G2GA|B2Ac BGG2:||:g2gb agfe|d2ed eggf| edec cega|bage d2dB|ABG A2GF|EGAB ADDA|[1 BcBc d2dB|c2cB AGED:||2 D2FA E2GE|DB, A,2 DB,A,B,|G,B,DB, G,2 BG|AFGA BGG2||

T:Gmajreel20
gabg gdBd|gbd'b c'abg|fadf agef|gdcB AAGD|{3ABA GB G2Bd|{3efg ab c'age|dega gdBd|gdcB cdef|gdec Bdgd|{3efg ab c'bag|bd'd'b c'2bc'|d'^c'af g2eb|gedB cdeg|dcBG gGcB|cBcd cAce|dedB cedc||

T:Gmajreel21
|: I/}de dega | bgag ea~~a2 | ge~e2 dega |baba ~b3a | ge~e2 geef | geee d~b3 | d'bab gedg | fdeg ed ||bdgb | agbg abag | fdef gebe | dBgd {3deg ag | ae3 ageg |1 bgab g2b2 :|2 aged efde |]

T:Gmajreel43
|: d |~c2 cc eAAc | ce c2 f2 dc | c2 ^dc edef | e2 fe ddBG |f2 fd dABc | defg fedc | B2 GB A2 A | dgfe deg2 | ddcB AGAB | g2 fe dBBd || e/2f/2e dc ecAe | d2 e2 ecBA | gedB GdcB | cdBA D2 A/2e/2e |f2 dc BdGG | gfga a2 ga | abgb a3 g | ea fd ecAB | c/2A/2A BG A2 Bc | [1 d2 BG eG{[:| [2 dBGB G2 :||

Fig. 20: Errores más comunes

Como se puede contemplar en la Fig 20, los errores más frecuentes son debidos a no completar el compás y no cerrar o abrir los caracteres que lo necesitan como () |: :| []. También se pueden apreciar errores sintácticos como |||::. Al contemplar los caracteres de dos en dos se aprecian más los fallos por que son más notorios cuando están mal.

6 Conclusiones

Tras realizar este trabajo llego a la conclusión de que la creación de arte artificial es algo real en la actualidad. Gracias a la computación se puede realizar arte de gran calidad por lo que, centrándonos en la música, para canciones folk se ha llegado a obtener muy buenos resultados. Asimismo, se han generando canciones indistinguibles entre si lo ha realizado una persona o una máquina para una persona que no tiene altos conocimientos sobre la música.
También se puede ver como especializar un data set para un estilo en concreto no parece mejorar los resultados. Esto es, las valoraciones son similares entrenando con el data set completo. Se contempla cómo los estilos se mezclan y, por este motivo, al comparar después las frecuencias de las notas hay bastantes diferencias entre un Jigs creado con un data set de sólo Jigs en Gmaj y otro generado por éste.

Un aspecto muy importante a destacar es el hardware ya que tiene que ser bastante actual y potente para poder realizar entrenamientos y creación musical. La memoria necesaria varía dependiendo del tamaño del batch y del vocabulario de entrada. Como consecuencia, por parte del tamaño del batch he visto que hay poca diferencia entre hacerlo de 8 o más. Así, si no se dispone de una tarjeta gráfica para hacer un batch size muy grande, a partir de 8 se podrían obtener buenos resultados. No obstante, el tiempo de entreno se vería penalizado contra más bajo mantengas el tamaño del batch. Cabe mencionar que el estudio estadístico no se ha realizado con la profundidad que hubiera requerido éste y no cumple con los p values estadísticos. Este hecho se debe a que conllevaría mucho tiempo y lo que he querido realizar es una pincelada a varios temas y no centrarme sólo en uno. Por lo tanto, con esto pretendo mostrar el camino para que se puedan realizar estudios más rigurosos.

Agradecimientos

Agradezco a toda la gente que me ha apoyado a realizar este trabajo, en especial a mi familia, a mi pareja Fifí, a Cristian, Alejandro y Julia por el apoyo final y a mi tutor Fernando Vilariño por la gran ayuda que me ha ofrecido.

Referències

[1] IRA KORSHUNOVA (2016). *Folk music style modelling using LSTMs* [en línea]. [consultado: 20 marzo 2018]. Disponible en Internet: https://github.com/IraKorshunova/folk-rnn.

[2] WALSHAW, Chris (2009). *abc music notation* [en línea]. [consultado: 22 marzo 2018]. Disponible en Internet:
abcnotation.com

[3] THE SESSION. *The Session* [en línea]. [consultado: 20 marzo 2018]. Disponible en Internet: https://thesession.org/

[4] LASAGNE. *Lasagne* [en línea]. [consultado: 22 marzo 2018]. Disponible en Internet: https://lasagne.readthedocs.io/en/latest/

[5] THEANO. *Theano*[en línea]. [consultado: 21 marzo 2018]. Disponible en Internet: http://deeplearning.net/software/theano/index.html

[6] ZAMORA, Erik (2017). *Redes Neuronales Recurrentes* [en línea].[Consultado 24/02/2018]. Disponible en Internet: https://es.scribd.com/doc/295974898/Redes-Neuronales-Recurrentes

[7] MATUK, Rosana (2017). *RNN y LSTM Redes Neuronales Profundas* [en línea]. DC-FCEyN-UBA, Segundo Cuatrimestre 2017.[Consultado 24/02/2018]. Disponible en Internet:
http://docplayer.es/76112611-Rnn-y-lstm-redes-neuronales-profundas.html

[8] JORDAN, Michael (1986). *Serial Order: a parallel distributed processing approach*.ICS Report 8604.

Institute for Cognitive Science, University of California, San Diego.

[9] KOUTNIK, Jan [et al.](2014). *A Clockwork RNN*. IDSIA, USI&SUPSI, Manno-Lugano, CH-6928, Switzerland
arXiv:1402.3511v1 [cs.NE] 14 Feb 2014

[10] O'BRIEN, Tim y ROMAN, Irán (2017). *A Recurrent Neural Network for Musical Structure Processing and Expectation*[en línea] Stanford University, Stanford, CA 94305 [Consultado 26/02/2018]. Disponible en Internet:
https://cs224d.stanford.edu/reports/O'BrienRom%C2%B4an.pdf

[11] MACDONALD, Kyle(2017).*Neural nets for generating music* [En línea][Consultado: 20/02/2018]. Disponible en Internet:
https://medium.com/artists-and-machine-intelligence/neural-nets-for-generating-music-f46dffac21c0

[12] VAN DEN OORD, Aäron [et al.] (2016) *WaveNet: A Generative Model For Raw Audio*. Google DeepMind, London, UK
arXiv: 1609.03499v1 [cs.SD] 12 de septiembre de 2016

[13] SKÚLI, Sigurður (2017). *How to Generate Music using a LSTM Neural Network in Keras* [En línea]. [consultado 12/03/2018] Disponible en Internet:
https://towardsdatascience.com/how-to-generate-music-using-a-lstm-neural-network-in-keras-68786834d4c5

[14] KARPATHY (2016). *char-rnn* [en línea]. [consultado: 05 de mayo 2018]. Disponible en Internet:
https://github.com/karpathy/char-rnn

[15] STRUM, B., SANTOS, J. F., & KORSHUNOVA, I. (2015).*Folk music style modelling by recurrent neural networks with long short term memory units.* 16th International Society for Music Information Retrieval Conference, late-breaking demo session. Presented at the 16th International Society for Music Information Retrieval Conference.

[16] L.STURM, Bob [et al.] (2016) *Music transcription modelling and composition using deep learning.* Centre for Digital Music, Queen Mary University of London arXiv:1604.08723v1 [cs.SD] 29 Apr 2016

[17] STURM, B., SANTOS, JF, y KORSHUNOVA, I. (2015). *Modelado de estilo de música folclórica por redes neuronales recurrentes con unidades de memoria a largo plazo.* 16ª Conferencia de la Sociedad Internacional de Recuperación de Información Musical, sesión demo demorada . Presentado en la 16ª Conferencia de la Sociedad Internacional de Recuperación de Información Musical.

[18] HUME, Colin. *Convert your own ABC to MIDI or PDF* [en línea]. [consultado: 22 mayo 2018]. Disponible en Internet: https://colinhume.com/music.aspx

[19] SANTIAGO, Manuel. (2018)*Encuesta Música Folk y Redes Neuronales* [en línea]. Disponible en Internet:
https://docs.google.com/forms/d/e/1FAIpQLSfV1tiHTwq9e8G5USZ5dlkoFQIiCK1sGzj_jsbr5Vx-4aSDFg/viewform?usp=sf_link

APÉNDICE

A.1 Notación anglosajona musical

Nomenclatura latina	Nomenclatura alfabética
DO	C
RE	D
MI	E
FA	F
SOL	G
LA	A
SI	B

TABLA 11: NOTACIÓN ANGLOSAJONA MUSICAL

A.2 Datos Cabecera ABC

X: 1 -> Numero de canción
T: The Legacy Jig -> Titulo de la canción
Z: Ramon -> Autor
S: www.sessiom.org -> Disponible en
M: 6/8 -> clave
L: 1/8 -> Duración
R: jig -> Estilo
K: Gmin -> Tonalidad

A.3 Evolución del Loss según Batch size

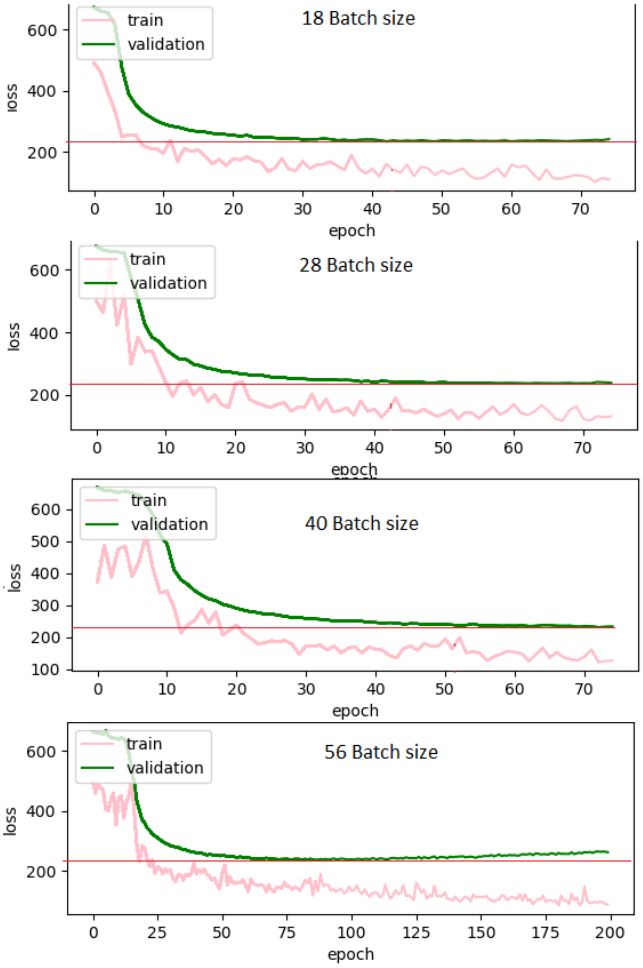


Fig. 21: Batch size

A.4 Cancion generado con el data set Jig Mayores a Gmaj

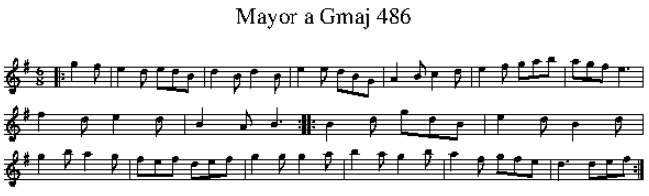


Fig. 22: Jig creado por la red con data set Mayor a Gmaj

```
X:486
T:Mayor a Gmaj 486
M:6/8
L:1/8
K:Gmaj
|: g2 f |e2 d edB | d2 B d2 B | e2 e dBG | A2 B c2 d |
e2 f gab | agf e3 | f2 d e2 d | B2 A B3 :||: B2 d gdB |
e2 d B2 d | g2 b a2 g | fef def |g2 g g2 a | b2 a g2 b |
a2 f gfe | d3 def :|
```

Fig. 23: ABC de la canción del pentagrama

A.5 Canción Real vs Canción Artificial

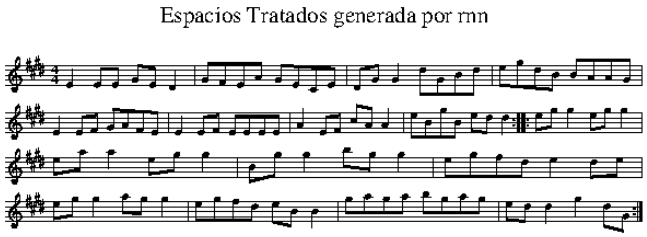


Fig. 24: Reel creado por la red con data set Espacios tratados

```
X:17
T:Espacios Tratados generada por rnn
M:4/4
K:Emaj
E2 EE GE D2|GFEA GECE|DG G2 dGBd|egdB BAAG| E2 EF GAFE|E2 EF EEEE|
A2 EF cA A2|eBgB ed d2:| |:eg g2 eg g2|ea a2 eg g2|Bg g2 bg g2|egfd e2 de|
eg g2 ag g2|egfd eB B2|gaga bgag|ed d2 g2 dG:|
```

Fig. 25: ABC de la canción del pentagrama

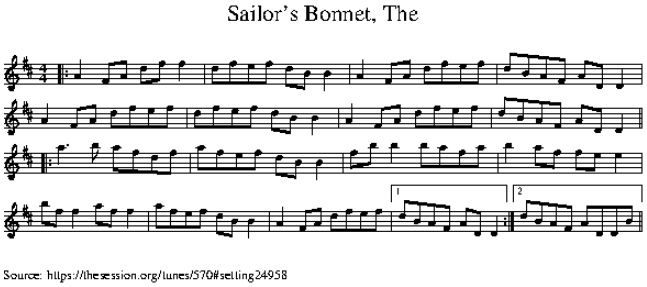


Fig. 26: Reel Original

A.6 Librerias y configuración

En este proyecto se utiliza Ubuntu 16.04, Python v2.7 y las siguientes librerias:

- Theano 1.0.1
- Lasagne 0.1
- Numpy 1.14.3
- CUDA 8.0 (para paralelizar por gpu)
- cudNN 8.0 (para paralelizar por gpu)
- libgpuarray 0.7.5 (para la generación de código GPU)
- BLAS contiene (lf77blas,latlas,lgfortran) (sintoniza automáticamente Álgebra Lineal Software, genérica estática)
- MKL 2018.0.2 (librería matemática)
- abc2midi (para crear midi a partir de abc)
- abc2abc (para transponer notas musicales en abc)

En la libreria lasagne hay un error en lasagne\layers\pool.py hay que cambiar la siguiente linea:
from theano.tensor.signal import downsample ->
from theano.tensor.signal import pool

El fichero de configuración .theanorc debe contener lo siguiente para la utilización de gpu:
$[cuda]$
root=/usr/local/cuda
$[global]$
floatX = float32
device = cuda0
set MKL_ THREADING_ LAYER = GNU
$[blas]$
ldflags = -lf77blas -latlas -lgfortran $[gpuarray]$
preallocate = 0.92
optimizer=fast_ compile

A.7 Hardware

El Hardware que se ha utilizzado para entrenar la red neuronal ha sido el siguiente:

- CPU: Intel Xeon E5 v4 3,5 GHz 4 nucleos 8 hilos
- GPU:Nvidia Titan Xp 12GB
- Memoria: DDR4 64GB

Para la generación de música:

- CPU: AMD Ryzen 2700x 8 nucleos 16 hilos
- GPU: Nvidia GTX 770 2GB
- Memoria: DDR4 16GB

A.8 Encuesta

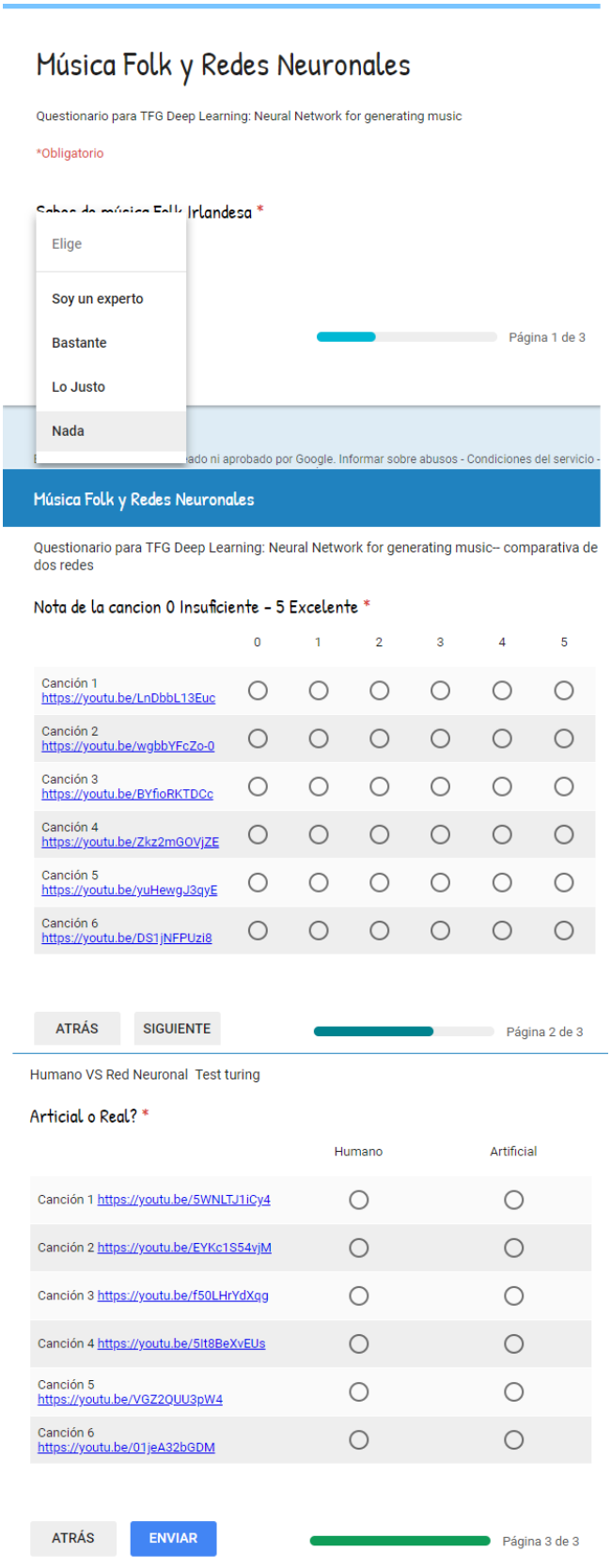


Fig. 27: Encuesta realizada para obtener los resultados

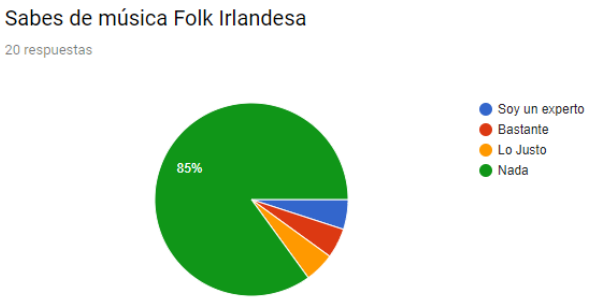


Fig. 28: Numero de participantes y nivel musical

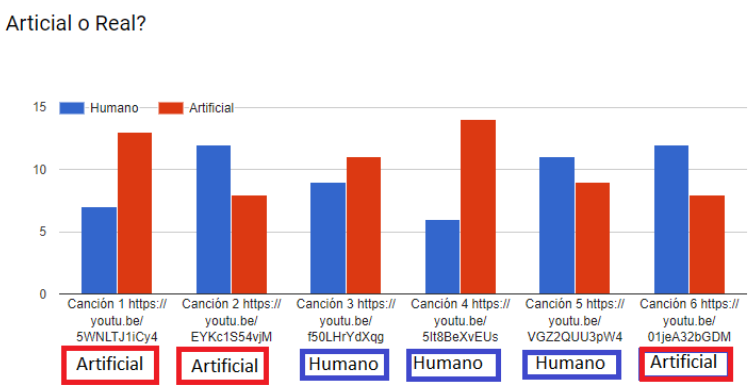


Fig. 29: Articial vs Humanos resultado de la encuesta

A.9 Pasos para generar MIDI ampliado

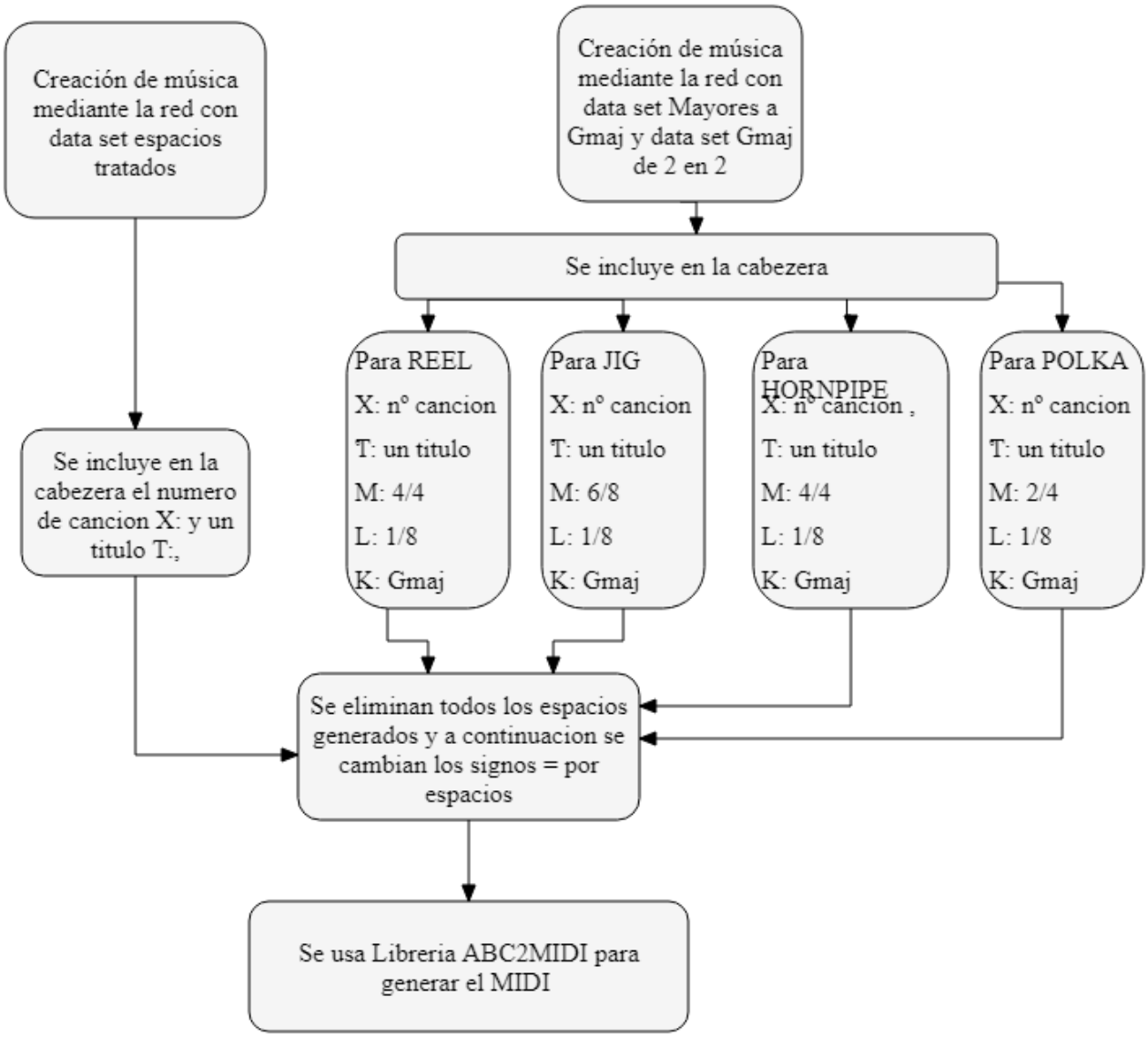


Fig. 30: Pasos para generar MIDI