Aprendizaje Automático

Generador de música

Integrantes:

- **Axel Martin Savizky**
- **Facundo Linari**
- **Joaquin Romera**
- **Jonathan Scherman**
- **Luciano Strika**

Generación de Música con Redes Neuronales Convolucionales

En este trabajo, replicamos lo hecho por Google Deep Mind en el paper WAVENET: A GENERATIVE MODEL FOR RAW AUDIO[1]

- Utilizamos un modelo autorregresivo basado en convoluciones causales dilatadas (dilated causal convolutions)
- Entrenado en un dataset de canciones en formato MIDI, logramos que genere canciones “plausibles” aunque enfrentamos límites de velocidad de convergencia dada la limitada infraestructura de la que disponemos.

Introducción: MIDI

Describimos a la música como una serie de eventos sonoros organizados de una manera determinada. Estos eventos pueden ser representados de diversas maneras; históricamente se perfeccionó la escritura musical mediante una notación simbólica en pentagramas, pero el soporte digital por excelencia es el protocolo MIDI, desarrollado a principios de 1980 para facilitar el uso entre instrumentos de diferentes fabricantes. El protocolo define mensajes que permiten identificar, entre otros eventos, las notas que son ejecutadas, el momento en que deben ejecutarse y durante cuánto tiempo.

Introducción: Por qué audio no.

Necesitamos una representación simbólica de piezas musicales para poder abstraernos de aspectos accidentales: son las notas ejecutadas en cierto orden lo que definen una pieza, no la ejecución en sí (lo que efectivamente registraría una grabación de audio).
Por otro lado, la simplicidad del formato MIDI hace que computacionalmente se requiera muchísimo menos trabajo que al trabajar con audio (ya que son menos las notas/tokens que si se usara una discretización directa de la onda de sonido).

Introducción: justificación

- El estudio de la interrelación entre aprendizaje automático y la generación de música podría permitir a compositores encontrar inspiración y asistirlos a desarrollar ideas.
- Desde un punto de vista comercial inclusive deberíamos poder generar música programáticamente, sin tener que incurrir en gastos de derechos de autor, por ejemplo.
- Puede ser de interés la interacción entre tecnología y arte, y como una influye en la otra.

Tareas Autorregresivas

En una tarea de autoregresión (autoregressive task) queremos, dada una secuencia de tokens S0...Sn, predecir el token S(n+1).

Aunque el ejemplo más conocido es language modelling (predecir la siguiente palabra/puntuación en un texto), esta task generaliza a otros problemas secuenciales, desde generación de imágenes (típicamente discretizadas en parches como en Visual Transformers[2]) hasta, más relevante para este trabajo, audio.

Dilated Causal Convolutions

WaveNet usa convoluciones causales para asegurarse de que el orden mediante el cual se modelan los datos se mantiene, esto es importante para las tareas de autoregresión. En particular las convoluciones *dilated* aplican un filtro para cubrir un área mayor, permitiendo a la red operar en campos más grandes utilizando menos capas.

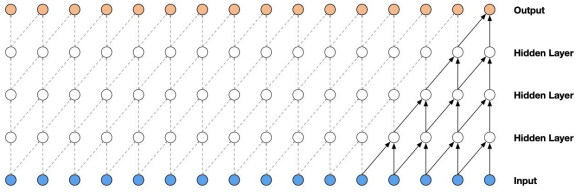


Figure 2: Visualization of a stack of causal convolutional layers.

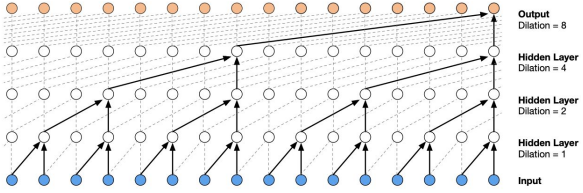


Figure 3: Visualization of a stack of *dilated* causal convolutional layers.

Nuestra tarea

En el marco de problemas autorregresivos, procesamos las melodías para dividirlas en secuencias de 33 notas/acordes, para luego ejecutar esta tarea semi-supervisada en la que nuestra red neuronal busca predecir, dadas 32 notas, la siguiente.

Datasets

Contamos con dos datasets:

- Uno pequeño, extraído junto a un artículo sobre WaveNet en el que basamos parte de la implementación.
- Uno significativamente más grande, provisto por Gwern.[3]

Este último sólo lo levantamos parcialmente, ya que el dataset entero tarda demasiado su preprocessing (aun usando paralelismo) pero estamos trabajando en eso.

Preprocesamiento

Solo se utilizan un conjunto reducido de instrumentos que tienden a usarse de forma monofónica y se transponen a Do mayor o La menor según corresponda para reducir la variabilidad de los datos. Además, no se tienen en cuenta las notas poco frecuentes para disminuir el ruido.

Ejemplo de archivo preprocesado

```
['A1' 'A1' 'A1' 'A1' 'A1' 'B1' 'C2' 'C#2' 'D2' 'C2' 'A1' 'C2' 'D2' 'C2'
 'A1' 'G1' 'A1' 'A1' 'E1' 'E1' 'A1' 'A1' 'E1' 'G1' 'E1' 'E1' 'B1' 'D2'
 'E1' 'E1' 'B1' 'D2' 'A1' 'G1' 'E1' 'G1' 'A1' 'A1' 'E1' 'G1' 'A1' 'A1'
 'A1' 'A1' 'A1' 'A1' 'A1' 'A1' 'A1' 'A1' 'A1' 'A1' 'A1' 'A1' 'B1' 'C2'
 'C#2' 'D2' 'C2' 'A1' 'C2' 'D2' 'C2' 'A1' 'G1' 'A1' 'A1' 'E1' 'E1' 'A1'
 'A1' 'E1' 'G1' 'E1' 'E1' 'B1' 'D2' 'E1' 'E1' 'B1' 'D2' 'A1' 'G1' 'E1'
 'G1' 'A1' 'A1' 'E1' 'G1' 'A1' 'C#2' 'E2' 'G1' 'A1' 'C#2' 'E2' 'G1' 'A1'
 'C#2' 'E2' 'G1' 'A1' 'G1' 'E1' 'G1' 'D2' 'C2' 'A1' 'C2' 'D2' 'C2' 'A1'
 'G1' 'A1' 'A1' 'E1' 'E1' 'A1' 'A1' 'E1' 'G1' 'E1' 'E1' 'B1' 'D2' 'E1'
 'E1' 'B1' 'D2' 'A1' 'G1' 'E1' 'G1' 'A1' 'A1' 'E1' 'G1' 'A1' 'A1' 'A1'
 'A1' 'A1' 'A1' 'A1' 'A1' 'A1' 'A1' 'A1' 'A1' 'A1' 'A1' 'B1' 'C2' 'C#2'
 'D2' 'C2' 'A1' 'C2' 'D2' 'C2' 'A1' 'G1' 'A1' 'A1' 'E1' 'E1' 'A1' 'A1'
 'E1' 'G1' 'E1' 'E1' 'B1' 'D2' 'E1' 'E1' 'B1' 'D2' 'A1' 'G1' 'E1' 'G1'
 'A1' 'A1' 'E1' 'G1' 'A1' 'C#2' 'E2' 'G1' 'A1' 'C#2' 'E2' 'G1' 'A1' 'C#2'
 'E2' 'G1' 'A1' 'G1' 'E1' 'G1' 'D2' 'C2' 'A1' 'C2' 'D2' 'C2' 'A1' 'G1'
 'A1' 'A1' 'E1' 'E1' 'A1' 'A1' 'E1' 'G1' 'E1' 'E1' 'B1' 'D2' 'E1' 'E1'
 'B1' 'D2' 'A1' 'G1' 'E1' 'G1' 'A1' 'A1' 'E1' 'G1' 'A1' 'A1' 'A1' 'A1'
 'A1' 'A1' 'A1' 'B1' 'C2' 'C#2' 'D2' 'C2' 'A1' 'C2' 'D2' 'C2' 'A1' 'G1'
 'A1' 'A1' 'E1' 'E1' 'A1' 'A1' 'E1' 'G1' 'E1' 'E1' 'B1' 'D2' 'E1' 'E1'
 'B1' 'D2' 'A1' 'G1' 'E1' 'G1' 'A1' 'A1' 'E1' 'G1' 'A1' 'C#2' 'E2' 'G1'
 'A1' 'C#2' 'E2' 'G1' 'A1' 'C#2' 'E2' 'G1' 'A1' 'G1' 'E1' 'G1' 'D2' 'C2'
 'A1' 'C2' 'D2' 'C2' 'A1' 'G1' 'A1' 'A1' 'E1' 'E1' 'A1' 'A1' 'E1' 'G1'
 'E1' 'E1' 'B1' 'D2' 'E1' 'E1' 'B1' 'D2' 'A1' 'G1' 'E1' 'G1' 'A1' 'A1'
 'A1' 'A1' 'A1']
```

Evaluación

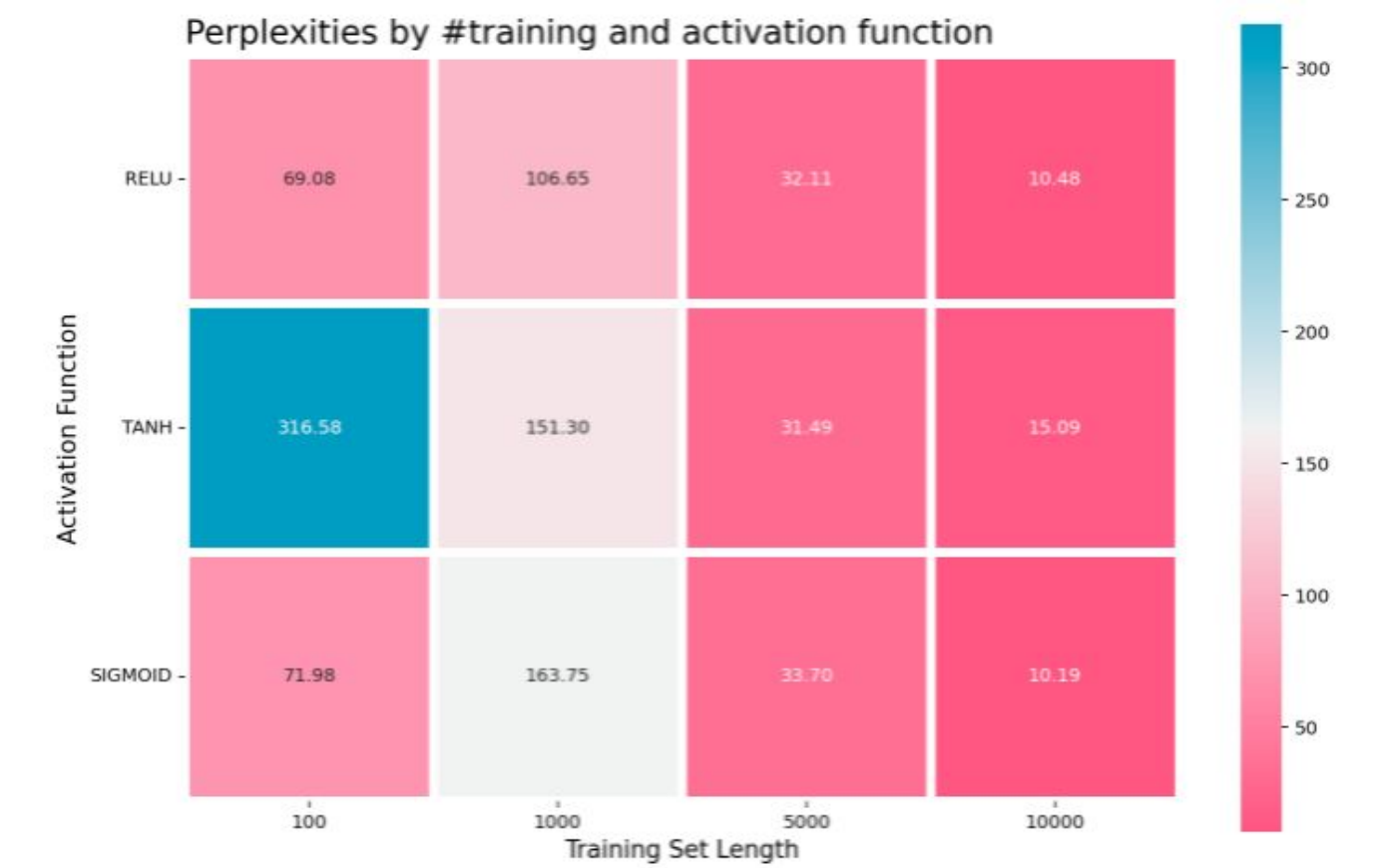
- Objetivo: entrenar un modelo de maximum-likelihood autorregresivo para poder generar melodías con algún grado de coherencia musical o plausibilidad.
- Además de evaluarlo cualitativamente (oir las melodías y juzgarlas) usaremos como métrica la perplejidad sobre datasets de test.
- **Perplejidad**: Dado un modelo generativo P(xn|x1...xn-1) y una muestra (una canción) m=m1...mn, queremos ver qué probabilidad P(m) le asigna el modelo.
  La perplejidad, dado un modelo P y sample m, es la media aritmética de las inversas de P(xi | x1...x(i-1)) para todo xi en m.

Perplejidad

$$PP(W) = \frac{1}{P(w_1, w_2, \ldots, w_N)^{\frac{1}{N}}}$$

$$= \sqrt[N]{\frac{1}{P(w_1, w_2, \ldots, w_N)}}$$

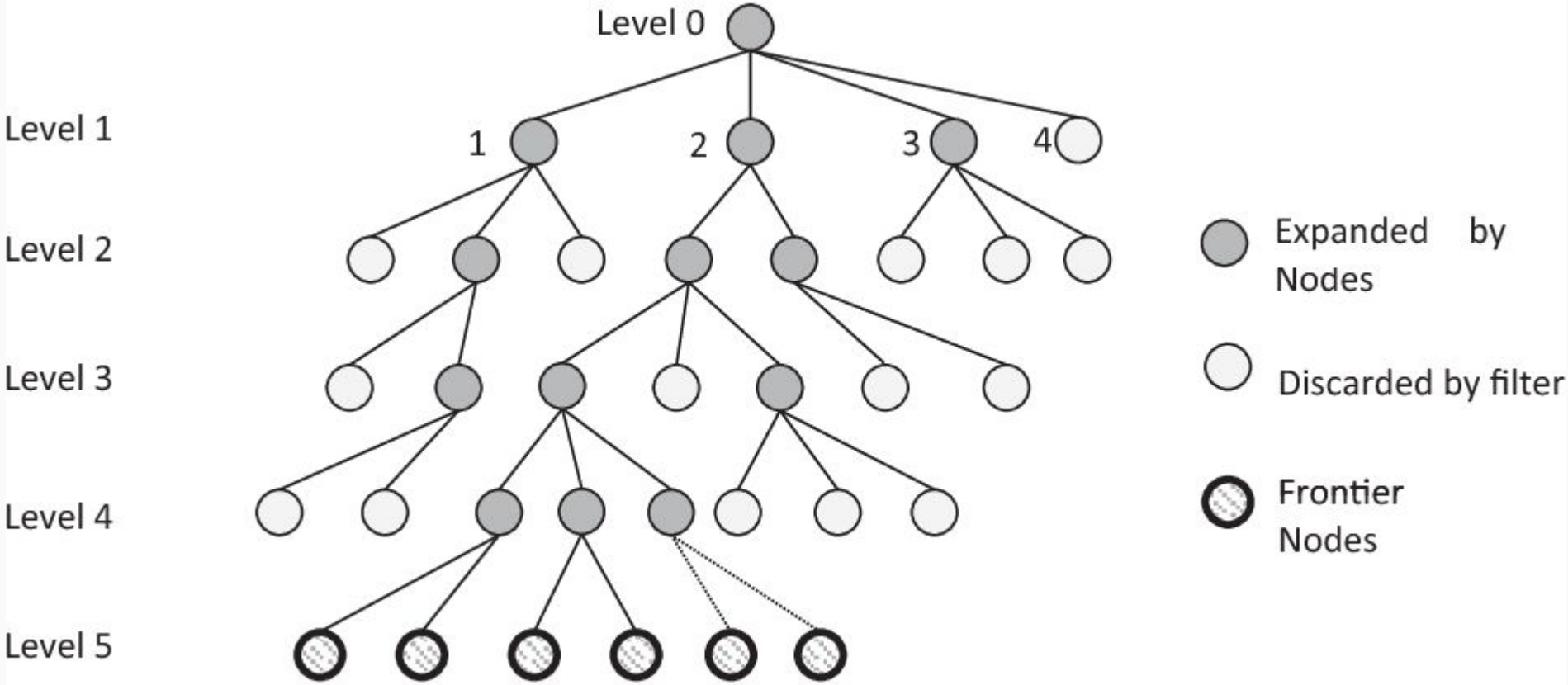
Resultados



Parámetros

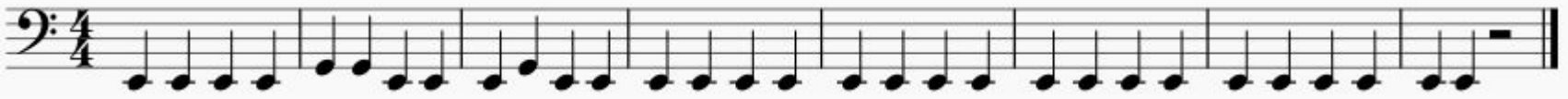
#test = 100

Generar melodías: Beam Search

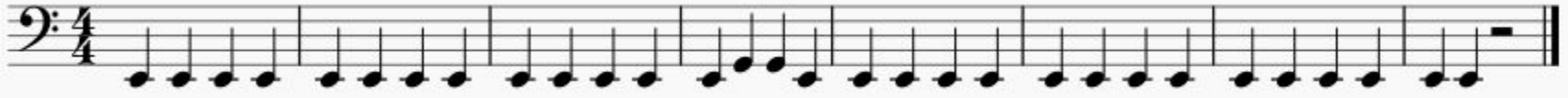


Melodías generadas:
Dataset de entrenamiento: 100 - Función de activación: tanh

k = 1

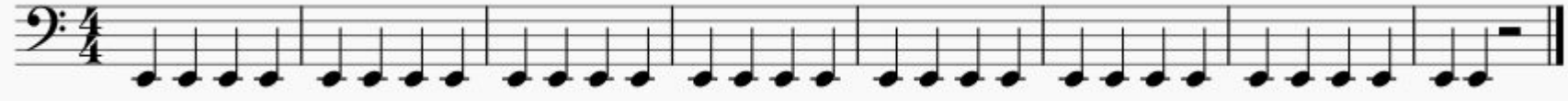


k = 100



Melodías generadas:
Dataset de entrenamiento: 10000 - Función de activación: tanh

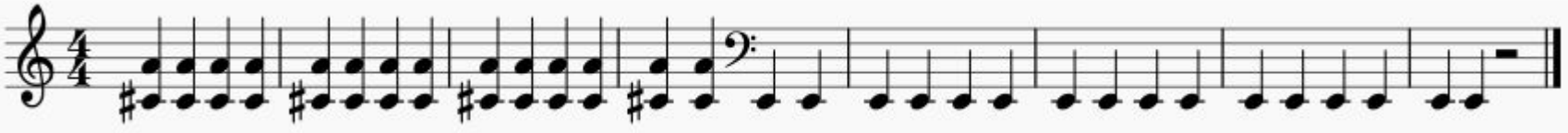
k = 1



k = 100

Melodías generadas:
Dataset de entrenamiento: 10000 - k = 100

Función de activación: relu



Función de activación: sigmoid

Heurísticas para generar mejores melodías

- Cantidad máxima de repeticiones: con un parámetro definir hasta cuantas veces se puede tocar la misma nota de forma seguida.

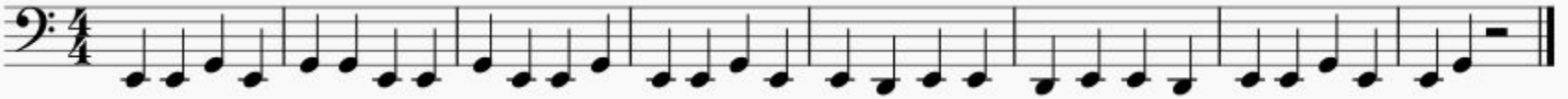
  DO DO ~~DO~~ RE DO DO ~~DO~~ RE DO...

- Lista tabú: prohibir que se pueda tocar una de las j últimas notas tocadas. j se pasa como parámetro.

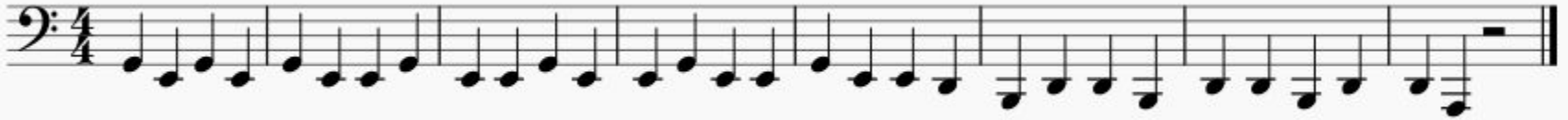
  DO ~~DO~~ RE ~~DO RE~~ MI DO ~~DO~~ RE...

Melodías generadas:
Máxima repetición: 2

Dataset de entrenamiento 100 - Función de activación: relu - k = 1



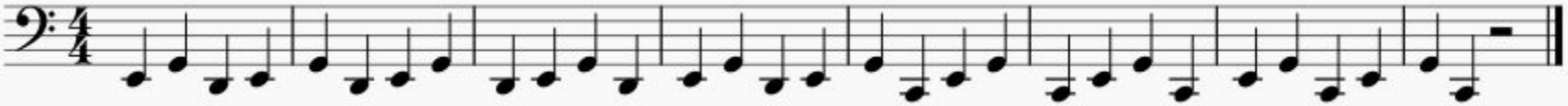
Dataset de entrenamiento 100 - Función de activación: relu - k = 100



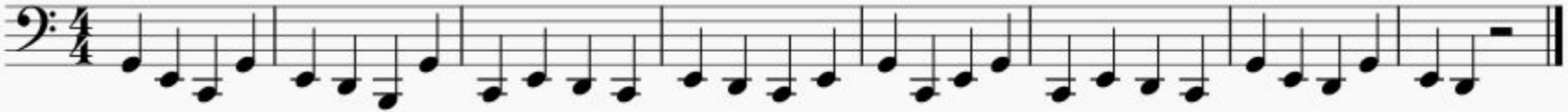
Dataset de entrenamiento 10000 - Función de activación: tanh - k = 100

Melodías generadas:
Dataset de entrenamiento: 100 - Función de activación: relu - Tamaño lista tabú: 2

k = 1

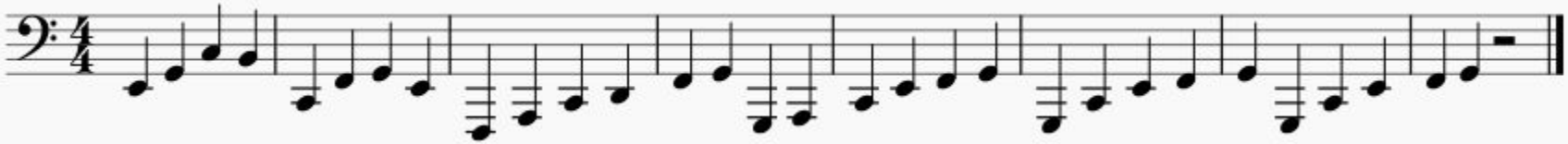


k = 100

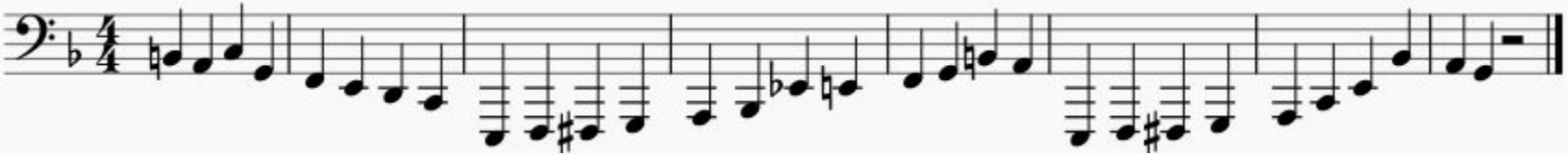


Melodías generadas:
Dataset de entrenamiento: 10000 - Función de activación: sigmoid - k: 1000

Tamaño lista tabú: 4



Tamaño lista tabú: 8



Análisis Beam Search

Mostrar perplejidad con beam search vs sin. Podemos comparar con varios n distintos

Referencias

1. Paper WAVENET: A GENERATIVE MODEL FOR RAW AUDIO: https://arxiv.org/pdf/1609.03499.pdf
2. Visual Transformers: https://arxiv.org/abs/2010.11929
3. Gwern: https://www.gwern.net/GPT-2-music