augmentation

January 10, 2022

[1]:

```
import librosa
import numpy as np

class Augmenter():

    @staticmethod
    def change_pitch(audio: any, sr: int, pitch_multiplier: int = 2, pitch_type:
 ↪ str = "down"):
        """Change the pitch of an audio file

        Parameters:
        audio (any): Audio file you want to change the pitch
        sr (int): Sample rate
        pitch_multiplier (int): The amount you want to change the pitch.␣
 ↪Default 2.
        pitch_type (str): if you want to change the pitch up or down

        Returns:
        Audio with changed pitch.
        """

        if pitch_type != "up" and pitch_type != "down":
            raise ValueError("pitch_type can only be 'up' or 'down'")

        y_pitch = audio.copy()
        bins_per_octave = 12
        pitch_pm = 2

        if pitch_type == "up":
            pitch_change = pitch_pm * pitch_multiplier
        else:
            pitch_change = pitch_pm * -pitch_multiplier

        y_pitch = librosa.effects.pitch_shift(
            y=y_pitch.astype('float64'),
            sr=sr,
            n_steps=pitch_change,
            bins_per_octave=bins_per_octave
        )

        return y_pitch

    @staticmethod
    def change_speed(audio: any, speed_change: str = "low",):
        """Change the speed of an audio file

        Parameters:
        audio (any): Audio file you want to change the speed
        speed_change (str): make the speed faster or slower
        Returns:
        Audio with changed speed.
        """

        if speed_change != "low" and speed_change != "high":
            raise ValueError("speed_change can only be 'low' or 'high'")

        y_speed = audio.copy()

        if speed_change == "low":
            change = 0.9
        else:
            change = 1.1

        tmp = librosa.effects.time_stretch(
            y_speed.astype('float64'), change)
        minlen = min(y_speed.shape[0], tmp.shape[0])
        y_speed *= 0
        y_speed[0:minlen] = tmp[0:minlen]

        return y_speed

    @staticmethod
    def change_speed_and_pitch(audio: any, sr: int, speed_change: str = "low", ␣
↪pitch_multiplier: int = 2, pitch_type: str = "down"):
        """change the speed and pitch of an audio file

        see change_pitch() and change_speed() for more documentation

        """

        if speed_change != "low" and speed_change != "high":
            raise ValueError("speed_change can only be 'low' or 'high'")

        if pitch_type != "up" and pitch_type != "down":
            raise ValueError("pitch_type can only be 'up' or 'down'")

        pitched_audio = Augmenter.change_pitch(
            audio, sr, pitch_multiplier=pitch_multiplier, pitch_type=pitch_type
        )

        speed_pitched_audio = Augmenter.change_speed(
            audio=pitched_audio, speed_change=speed_change)

        return speed_pitched_audio

    @staticmethod
    def hpss(audio: any):
        y_hpss = lb.effects.hpss(audio.astype('float64'))
        return y_hpss

    @staticmethod
    def value_augmentation(audio: any):
        """Stretches the length of the audio

        Parameters:
        audio:any : Any audio file you'd like to use

        Returns:
        Audio with augmented values"""
        y_aug = audio.copy()
        dyn_change = np.random.uniform(low=1.5,high=3)
        y_aug = y_aug * dyn_change
        return y_aug

    @staticmethod
    def add_distribution_noise(audio: any):
        """Add distribution noise to the audio file

        Parameters:
        audio:any : Any audio file you'd like to use

        Returns:
        Audio with distribution noise."""
        y_noise = audio.copy()
        noise_amp = 0.005*np.random.uniform()*np.amax(y_noise)
        y_noise = y_noise.astype('float64') + noise_amp * np.random.
↪normal(size=y_noise.shape[0])
        return y_noise
```

[ ]:

```

```