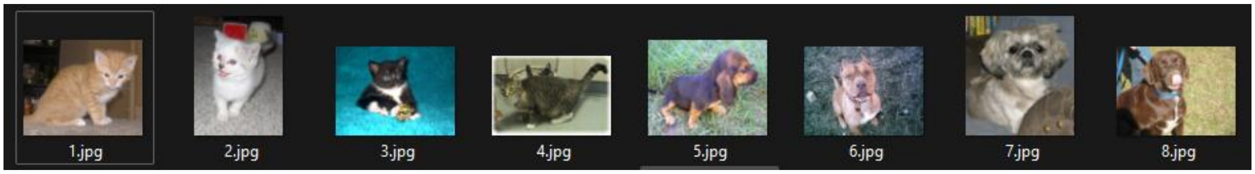
6^{η} εργασία Deep Learning

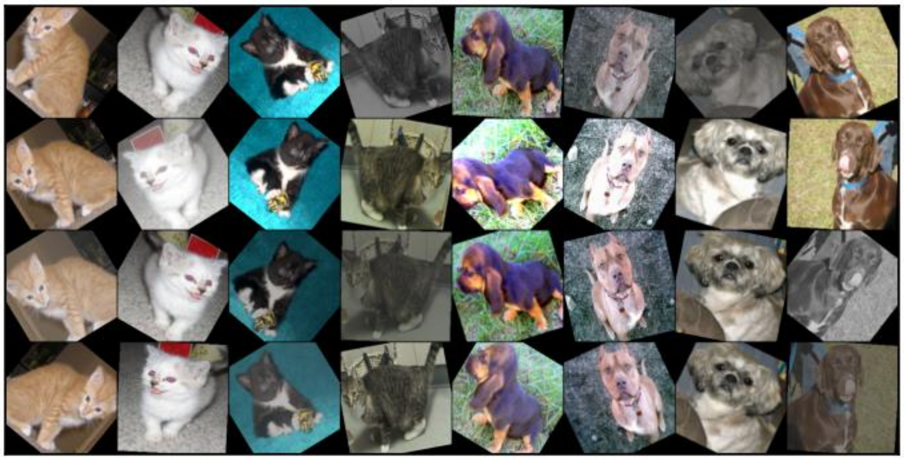
Κώδικας Augmentation για εικόνες:

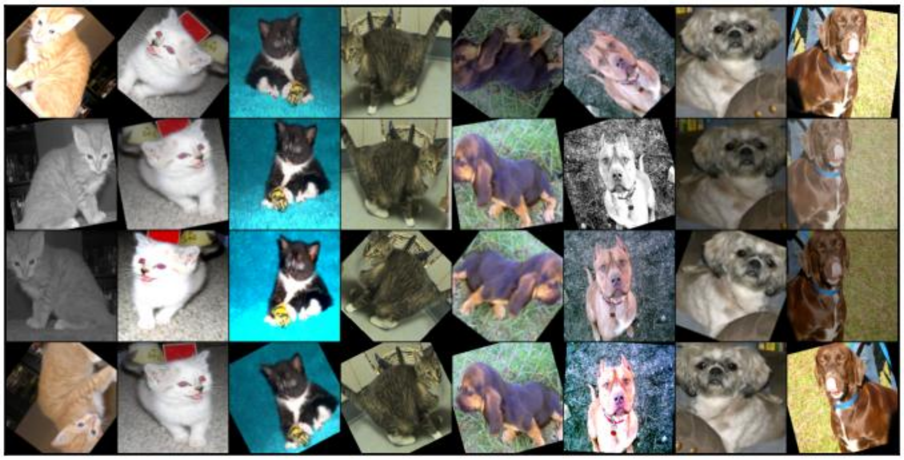
```
1.
import torch
from torchvision.transforms import transforms
from torch.utils.data import Dataset
import pandas as pd
import os
from skimage import io,transform
import matplotlib.pyplot as plt
from torchvision.utils import save_image
from torchvision.utils import make_grid
import torchvision.transforms.functional as F
import numpy as np



def show(imgs):
    if not isinstance(imgs, list):
        imgs = [imgs]
    fix, axs = plt.subplots(ncols=len(imgs), squeeze=False)
    for i, img in enumerate(imgs):
        img = img.detach()
        img = F.to_pil_image(img)
        axs[0, i].imshow(np.asarray(img))
        axs[0, i].set(xticklabels=[], yticklabels=[], xticks=[], yticks=[])

class CustomDataset(Dataset):
    def __init__(self,csv_file,transform = None):
        self.transform = transform
        self.data = pd.read_csv(csv_file)

    def __len__(self):
        return len(self.data)

    def __getitem__(self,index):

        name = os.path.join('data', self.data.iloc[index,0])
        image = io.imread(name)
        #image = Image.fromarray(image)

        y_label = torch.tensor(int(self.data.iloc[index,1]))

        if self.transform:
            image = self.transform(image)

        return (image,y_label)


transform = transforms.Compose([
    transforms.ToPILImage(),
    transforms.Resize((256,256)),
    transforms.RandomCrop((224,224)),
    transforms.RandomHorizontalFlip(p=0.5),
    transforms.RandomVerticalFlip(p=0.05),

transforms.RandomApply([transforms.ColorJitter(brightness=0.5,contrast=0.5,saturation=0.5)],p=0.5),
    transforms.RandomRotation(45),
    transforms.RandomGrayscale(0.1),
    transforms.RandomApply([transforms.GaussianBlur(3)],p=0.5),
    transforms.ToTensor()
   # transforms.Normalize([0.5,0.5,0.5], [0.5,0.5,0.5])
    ])

dataset = CustomDataset('data/info.csv',transform = transform)
img_num = 9
df = pd.read_csv('data/info.csv')
a = []


for _ in range(10):
    for img, label in dataset:
        a.append(img)
        path = 'data/'+str(img_num)+'.jpg'
        save_image(img, path)
        df.loc[img_num] = (str(img_num)+'.jpg',label.item())

        img_num += 1

df.to_csv('data/info.csv',index=False)
show(make_grid(a[:32]))
```

Αρχικές εικόνες (8 εγγραφές):



Augmentation:

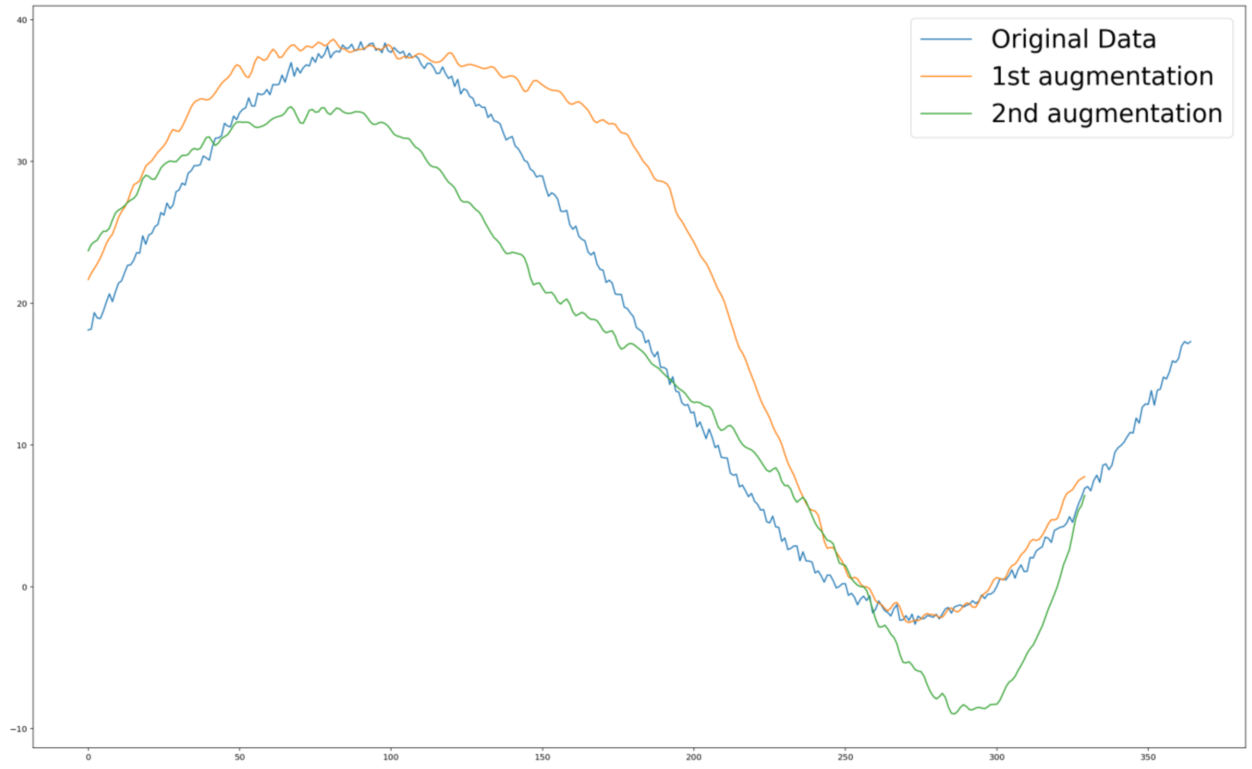


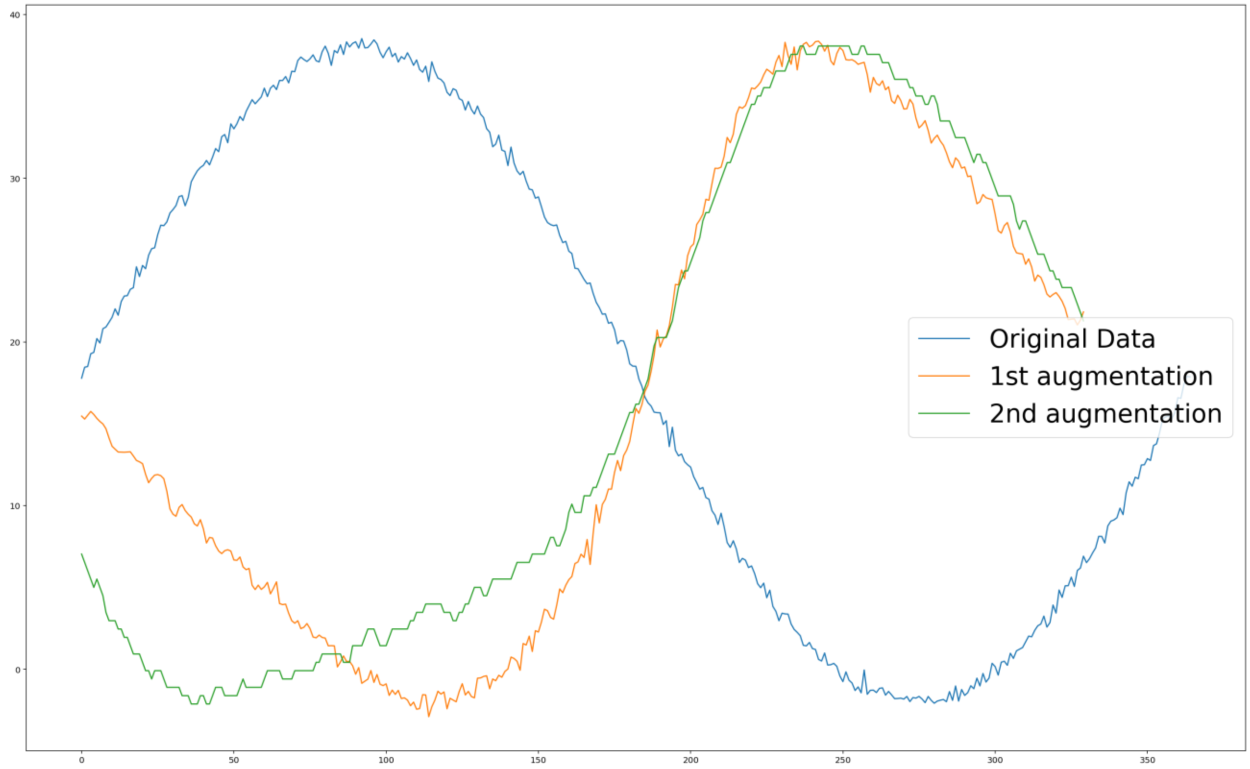


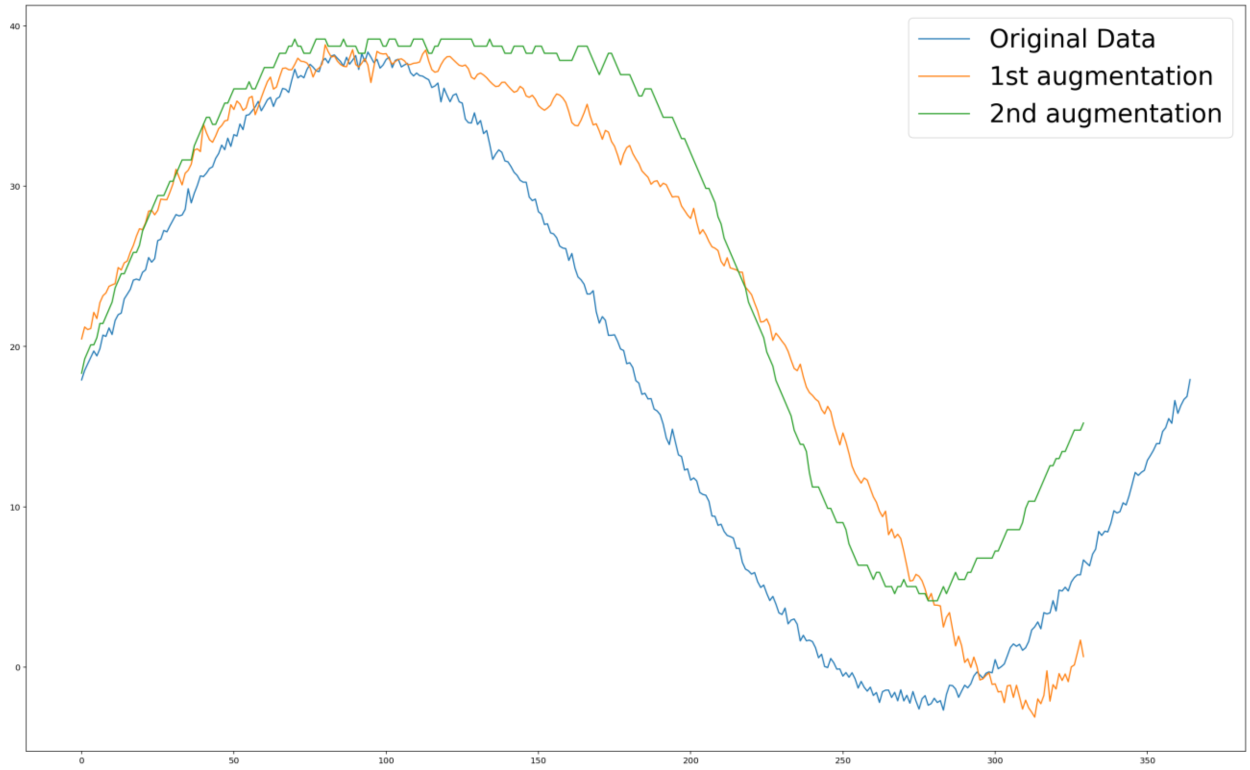
Κώδικας Time series:

```
1.
import numpy as np
import pandas as pd
import random
import matplotlib.pyplot as plt
from tsaug import TimeWarp, Crop, Quantize, Drift, Reverse,AddNoise,Convolve
from tsaug.visualization import plot

days = 365
t = np.arange(days)

temps = 10 * np.sin(2 * np.pi * t / days)
avg_temp = 18 #GLOBAL WARMING!!
data = avg_temp + temps
noise = np.random.normal(0,0.3,size = days)

original = temps + data + noise
#original = original.reshape(1,-1)


augmenter = (
        AddNoise(scale=0.01)    #add small noise in the data
        + TimeWarp()*5      #compute 5 times in parallel all the transforms bellow
        + Convolve(window= 'flattop',size =11,prob=0.8)     #smoothen with kernel flattop
        + Crop(size= 330) #Crop the computed data to introduce some steep slopes
        + Drift(max_drift=0.2,n_drift_points=5,prob=0.5)
        + Quantize(n_levels=80,prob=0.2)
        + Reverse(prob= 0.2)
    )

y = augmenter.augment(original)

new = np.concatenate(y)

plt.figure(figsize=(26,16))
plt.plot(original)
plt.plot(y[0])
plt.plot(y[1])
#plt.plot(new)
plt.legend(['Original Data','1st augmentation','2nd augmentation'])
plt.show()
```

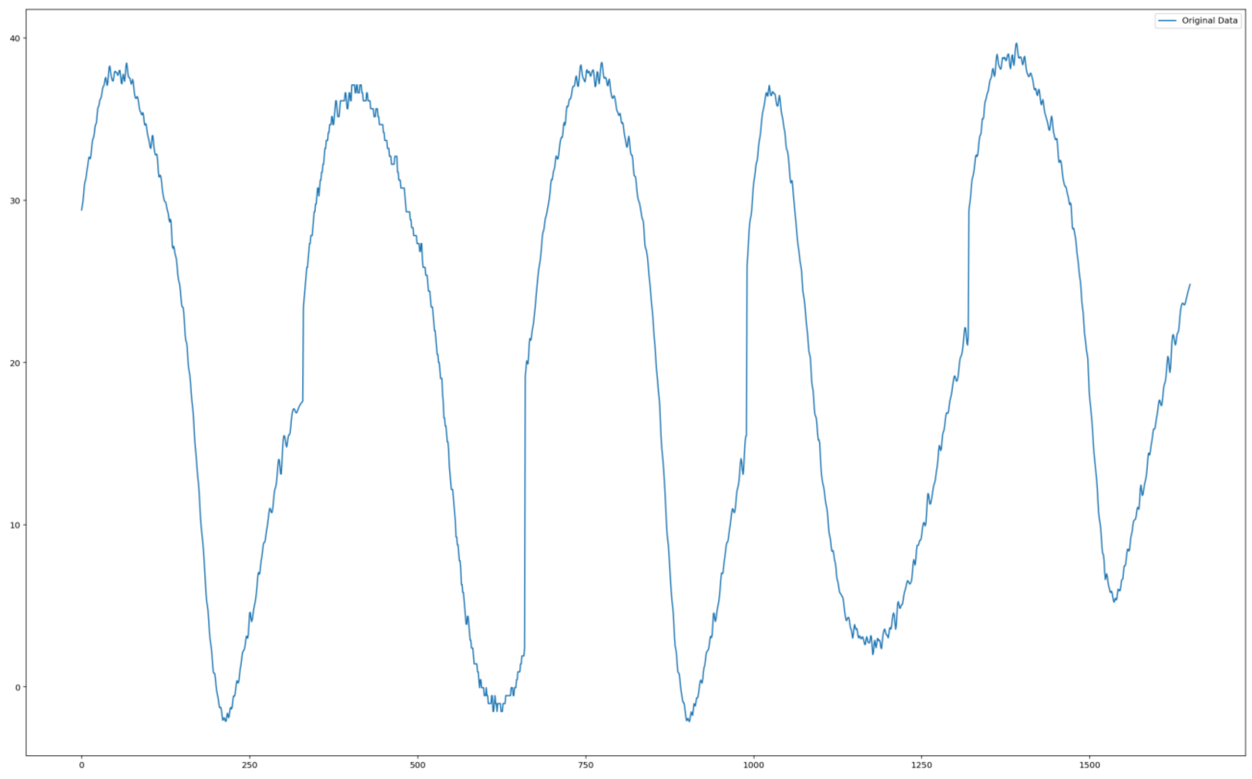
Αποτελέσματα:

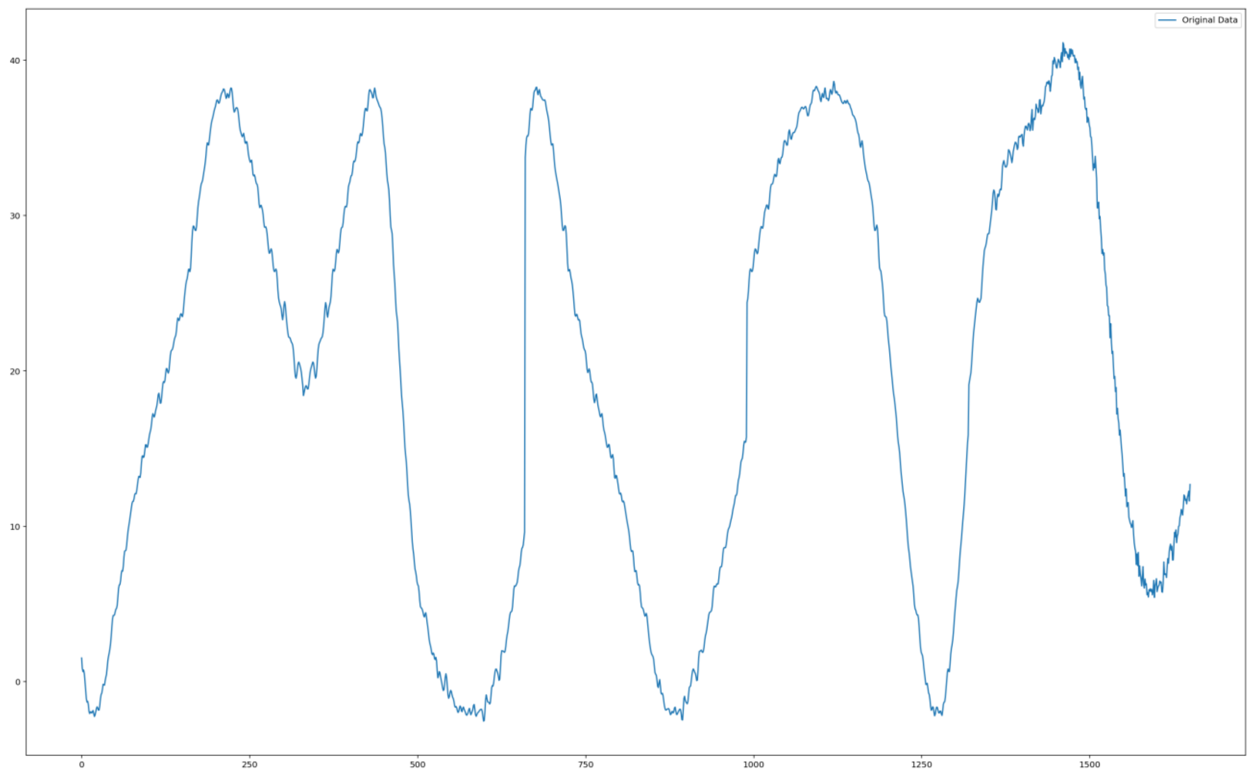






Όλα τα υπολογισμένα augmentation μαζί:


Original Data
40
30
20
10
0
0
250
500
750
1000
1250
1500


Original Data
40
30
20
10
0
0
250
500
750
1000
1250
1500