Spoken Language Processing

Homework 1

1. Ссылка на занятие: https://www.twitch.tv/deeplearningschool
2. Материалы: https://vk.cc/bVLCOC https://vk.cc/bVLD3Z

В этом задании предлагается обучить классификатор класса возраста по голосу (пример с тем, как это можно сделать для пола см. в семинаре)

P.S. не забудьте, что если то вы работает в Colab, то вы можете поменять среду выполнения на GPU/TPU!

Вопросы по заданию/материалам: @Nestyme

```
1 !pip3 install timit-utils==0.9.0
```

```
Collecting timit-utils==0.9.0
  Downloading https://files.pythonhosted.org/packages/22/32/0c98f7f44386947b9e4080f54f09a7380c390e0b8337ab0b87050d49c43a/timit_
Requirement already satisfied: matplotlib in /usr/local/lib/python3.6/dist-packages (from timit-utils==0.9.0) (3.2.2)
Collecting python-speech-features
  Downloading https://files.pythonhosted.org/packages/ff/d1/94c59e20a2631985fbd2124c45177abaa9e0a4eee8ba8a305aa26fc02a8e/python
Requirement already satisfied: numpy in /usr/local/lib/python3.6/dist-packages (from timit-utils==0.9.0) (1.19.4)
Requirement already satisfied: scipy in /usr/local/lib/python3.6/dist-packages (from timit-utils==0.9.0) (1.4.1)
Requirement already satisfied: pandas in /usr/local/lib/python3.6/dist-packages (from timit-utils==0.9.0) (1.1.5)
Collecting SoundFile>=0.8.0
  Downloading https://files.pythonhosted.org/packages/eb/f2/3cbbbf3b96fb9fa91582c438b574cff3f45b29c772f94c400e2c99ef5db9/SoundF
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.6/dist-packages (from matplotlib->timit-utils==0.9.0
Requirement already satisfied: python-dateutil>=2.1 in /usr/local/lib/python3.6/dist-packages (from matplotlib->timit-utils==0.
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in /usr/local/lib/python3.6/dist-packages (from matplot
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.6/dist-packages (from matplotlib->timit-utils==0.9.0) (0.
Requirement already satisfied: pytz>=2017.2 in /usr/local/lib/python3.6/dist-packages (from pandas->timit-utils==0.9.0) (2018.9
Requirement already satisfied: cffi>=1.0 in /usr/local/lib/python3.6/dist-packages (from SoundFile>=0.8.0->timit-utils==0.9.0)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.6/dist-packages (from python-dateutil>=2.1->matplotlib->timit
Requirement already satisfied: pycparser in /usr/local/lib/python3.6/dist-packages (from cffi>=1.0->SoundFile>=0.8.0->timit-uti
Building wheels for collected packages: python-speech-features
  Building wheel for python-speech-features (setup.py) ... done
  Created wheel for python-speech-features: filename=python_speech_features-0.6-cp36-none-any.whl size=5890 sha256=af11ec629012
```

```
  Stored in directory: /root/.cache/pip/wheels/3c/42/7c/f60e9d1b40015cd69b213ad90f7c18a9264cd745b9888134be
Successfully built python-speech-features
Installing collected packages: python-speech-features, SoundFile, timit-utils
Successfully installed SoundFile-0.10.3.post1 python-speech-features-0.6 timit-utils-0.9.0
```

```
1 !pip3 install torchaudio
```

```
Collecting torchaudio
  Downloading https://files.pythonhosted.org/packages/2a/f9/618434cf4e46dc975871e1516f5499abef6564ab4366f9b2321ee536be14/torcha
     |████████████████████████████████| 7.6MB 8.0MB/s
Collecting torch==1.7.1
  Downloading https://files.pythonhosted.org/packages/90/4f/acf48b3a18a8f9223c6616647f0a011a5713a985336088d7c76f3a211374/torch-
     |████████████████████████████████| 776.8MB 22kB/s
Requirement already satisfied: numpy in /usr/local/lib/python3.6/dist-packages (from torch==1.7.1->torchaudio) (1.19.4)
Requirement already satisfied: typing-extensions in /usr/local/lib/python3.6/dist-packages (from torch==1.7.1->torchaudio) (3.7
Requirement already satisfied: dataclasses; python_version < "3.7" in /usr/local/lib/python3.6/dist-packages (from torch==1.7.1
ERROR: torchvision 0.8.1+cu101 has requirement torch==1.7.0, but you'll have torch 1.7.1 which is incompatible.
Installing collected packages: torch, torchaudio
  Found existing installation: torch 1.7.0+cu101
    Uninstalling torch-1.7.0+cu101:
      Successfully uninstalled torch-1.7.0+cu101
Successfully installed torch-1.7.1 torchaudio-0.7.2
```

```
1 ! wget https://ndownloader.figshare.com/files/10256148
```

```
--2020-12-27 08:45:27--  https://ndownloader.figshare.com/files/10256148
Resolving ndownloader.figshare.com (ndownloader.figshare.com)... 34.248.245.115, 34.240.75.121, 3.248.141.196, ...
Connecting to ndownloader.figshare.com (ndownloader.figshare.com)|34.248.245.115|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://s3-eu-west-1.amazonaws.com/pfigshare-u-files/10256148/TIMIT.zip [following]
--2020-12-27 08:45:28--  https://s3-eu-west-1.amazonaws.com/pfigshare-u-files/10256148/TIMIT.zip
Resolving s3-eu-west-1.amazonaws.com (s3-eu-west-1.amazonaws.com)... 52.218.97.10
Connecting to s3-eu-west-1.amazonaws.com (s3-eu-west-1.amazonaws.com)|52.218.97.10|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 440207227 (420M) [binary/octet-stream]
Saving to: ‘10256148’

10256148            100%[===================>] 419.81M  21.7MB/s    in 21s
```

```
2020-12-27 08:45:49 (20.4 MB/s) - ‘10256148’ saved [440207227/440207227]
```

```
!unzip -q 10256148
```

```
import timit_utils as tu
import os
import librosa
import numpy as np
from tqdm import tqdm

import torch
import torch.nn as nn
from torch.optim import Adam
import torch.nn.functional as F

import matplotlib.pyplot as plt
from sklearn.metrics import accuracy_score

import IPython
_TIMIT_PATH = 'data/lisa/data/timit/raw/TIMIT'
```

Задание 1

Загрузите данные для обучения. Для этого:

1. Скачайте датасет TIMIT (см семинар)
2. Соберите пары "голос" — "класс возраста" также, как на семинаре собирались пары "голос" — "пол". Аудиодорожки сконвертируйте в мелспектрограммы при помощи `torchaudio либо librosa`

P.S. вы можете использовать свою реализацию, а можете предложенную (см следующие ячейки)

```
import timit_utils as tu
```

```
import os
import librosa
import numpy as np
from tqdm import tqdm
import torch as t


class timit_dataloader:
    def __init__(self, data_path=_TIMIT_PATH, train_mode=True, age_mode=True):
        self.doc_file_path = os.path.join(data_path, 'DOC', 'SPKRINFO.TXT')
        self.corpus = tu.Corpus(data_path)
        with open(self.doc_file_path) as f:
            self.id_age_dict = dict(
                [(tmp.split(' ')[0], 86 - int(tmp.split('  ')[5].split('/')[-1].replace('??', '50'))) \
                 for tmp in f.readlines()[39:]])
        if train_mode:
            self.trainset = self.create_dataset('train', age_mode=age_mode)
            self.validset = self.create_dataset('valid', age_mode=age_mode)
        self.testset = self.create_dataset('test', age_mode=age_mode)

    def return_age(self, id):
        return self.id_age_dict[id]

    def return_data(self):
        return self.trainset, self.validset, self.testset

    def return_test(self):
        return self.testset

    def create_dataset(self, mode, age_mode=False):
        global people
        assert mode in ['train', 'valid', 'test']
        if mode == 'train':
            people = [self.corpus.train.person_by_index(i) for i in range(350)]
        if mode == 'valid':
            people = [self.corpus.train.person_by_index(i) for i in range(350, 400)]
        if mode == 'test':
            people = [self.corpus.test.person_by_index(i) for i in range(150)]
```

```
        people = [self.corpus.test.person_by_index(i) for i in range(150)]
        spectrograms_and_targets = []
        for person in tqdm(people):
            try:
                target = self.return_age(person.name)
                for i in range(len(person.sentences)):
                    spectrograms_and_targets.append(
                        self.preprocess_sample(person.sentence_by_index(i).raw_audio, target, age_mode=True))
            except:
                print(person.name, target)

        X, y = map(np.stack, zip(*spectrograms_and_targets))
        X = X.transpose([0, 2, 1])  # to [batch, time, channels]
        return X, y

    @staticmethod
    def spec_to_image(spec, eps=1e-6):
        mean = spec.mean()
        std = spec.std()
        spec_norm = (spec - mean) / (std + eps)
        spec_min, spec_max = spec_norm.min(), spec_norm.max()
        spec_scaled = 255 * (spec_norm - spec_min) / (spec_max - spec_min)
        spec_scaled = spec_scaled.astype(np.uint8)
        return spec_scaled

    @staticmethod
    def clasterize_by_age(age):
        if age <= 25:
            return 0
        if 25 < age <= 40:
            return 0.5
        if age > 40:
            return 1

    def preprocess_sample(self, amplitudes, target=None, age_mode=False, sr=16000, max_length=150):
        spectrogram = librosa.feature.melspectrogram(amplitudes, sr=sr, n_mels=128, fmin=1, fmax=8192)[:, :max_length]
        spectrogram = np.pad(spectrogram, [[0, 0], [0, max(0, max_length - spectrogram.shape[1])]], mode='constant')
        if target == None:
```

```
          return np.array([self.spec_to_image(np.float32(spectrogram))]).transpose([0, 2, 1])
        target = self.clasterize_by_age(target)
        return self.spec_to_image(np.float32(spectrogram)), target

    def preprocess_sample_inference(self, amplitudes, sr=16000, max_length=150, device='cpu'):
        spectrogram = librosa.feature.melspectrogram(amplitudes, sr=sr, n_mels=128, fmin=1, fmax=8192)[:, :max_length]
        spectrogram = np.pad(spectrogram, [[0, 0], [0, max(0, max_length - spectrogram.shape[1])]], mode='constant')
        spectrogram = np.array([self.spec_to_image(np.float32(spectrogram))]).transpose([0, 2, 1])

        return t.tensor(spectrogram, dtype=t.float).to(device, non_blocking=True)


class dataloader:
    def __init__(self, spectrograms, targets):
        self.data = list(zip(spectrograms, targets))

    def next_batch(self, batch_size, device):
        indices = np.random.randint(len(self.data), size=batch_size)

        input = [self.data[i] for i in indices]

        source = [line[0] for line in input]
        target = [line[1] for line in input]

        return self.torch_batch(source, target, device)

    @staticmethod
    def torch_batch(source, target, device):
        # print('source',source)
        # print('target',target)
        return tuple(
            [
                t.tensor(val, dtype=t.float).to(device, non_blocking=True)
                for val in [source, target]
            ]
        )

    @staticmethod
```

```
    def padd_sequences(lines, pad_token=0):
        lengths = [len(line) for line in lines]
        max_length = max(lengths)

        return np.array(
            [
                line + [pad_token] * (max_length - lengths[i])
                for i, line in enumerate(lines)
            ]
        )
```

Простая сверточная сеть, ее можно дотюнить или поменять по желанию

```

import torch
import torch.nn as nn
import torch.nn.functional as F


class Model(nn.Module):
    def __init__(self, window_sizes=(3, 4, 5)):
        super(Model, self).__init__()

        self.convs = nn.ModuleList([
            nn.Conv2d(1, 128, [window_size, 128], padding=(window_size - 1, 0))
            for window_size in window_sizes
        ])

        self.fc = nn.Linear(128 * len(window_sizes), 1)

    def forward(self, x):
        # Apply a convolution + max pool layer for each window size
        x = torch.unsqueeze(x, 1)  # [B, C, T, E] Add a channel dim.
        xs = []
        for conv in self.convs:
            x2 = F.relu(conv(x))  # [B, F, T, 1]
```

```
            x2 = torch.squeeze(x2, -1)  # [B, F, T]
            x2 = F.max_pool1d(x2, x2.size(2))  # [B, F, 1]
            xs.append(x2)
        x = torch.cat(xs, 2)  # [B, F, window]

        # FC
        x = x.view(x.size(0), -1)  # [B, F * window]
        logits = self.fc(x)  # [B, class]
        probs = torch.sigmoid(logits).view(-1)
        return probs

    def loss(self, probs, targets):
        return nn.BCELoss()(probs.float(), targets.float())
```

```
device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')
print(f'using {device} mode')
patience = 500
best_loss = 1000
cnt = 0

model = Model()
if device == torch.device('cuda'):
    model.cuda()
else:
    model.cpu()
model.train()
```

```
using cpu mode
Model(
  (convs): ModuleList(
    (0): Conv2d(1, 128, kernel_size=[3, 128], stride=(1, 1), padding=(2, 0))
    (1): Conv2d(1, 128, kernel_size=[4, 128], stride=(1, 1), padding=(3, 0))
    (2): Conv2d(1, 128, kernel_size=[5, 128], stride=(1, 1), padding=(4, 0))
  )
  (fc): Linear(in_features=384, out_features=1, bias=True)
)
```

```
_timit_dataloader = timit_dataloader()
train, valid, test = _timit_dataloader.return_data()

trainset = dataloader(*train)
validset = dataloader(*valid)
testset = dataloader(*test)
BATCH_SIZE = 64

optimizer = Adam(
    [p for p in model.parameters() if p.requires_grad], betas=(0.9, 0.999), eps=1e-5
)
```

```
100%|██████████| 350/350 [00:42<00:00,  8.28it/s]
100%|██████████| 50/50 [00:06<00:00,  8.28it/s]
100%|██████████| 150/150 [00:18<00:00,  8.03it/s]
```

Задание 2

1. Обучите свой классификатор категории возраста
2. Попробуйте улучшить результат. Можно попробовать усложнить сетку, подвигать границы категорий, поискать новые данные, что угодно, кроме учиться на тесте :)

Чем больше границы тем лучше качество , чем меньше границы тем хуже качество.

3. Какой подход оказался самым эффективным? Как думаете, почему?

Эффективней всего обучать маленьких границах но при этом предсказывать более большие тогда качество должно повыситься Возможно вобще надо предсказывать в днях или месяцах ,а потом переводить в года. Не успел проверить.

4. Как считаете, где можно было бы применить такой классификатор в качестве вспомогательной задачи?

Как вспомогательная задача может быть применима в голосовых помощниках, или умных домах

```
import torch as t
for i in tqdm(range(100)):
```

```
    optimizer.zero_grad()
    # print('batch',BATCH_SIZE,'device',device)
    input, target = trainset.next_batch(BATCH_SIZE, device=device)
    out = model(input)
    loss = model.loss(out, target)
    loss.backward()
    optimizer.step()

    if i % 50 == 0:
        model.eval()

        with torch.no_grad():
            optimizer.zero_grad()

            input, target = validset.next_batch(BATCH_SIZE, device=device)
            out = model(input)
            valid_loss = model.loss(out, target)
            out, target = out.cpu().detach().numpy(), target.cpu().detach().numpy()
            # print(out, target)
            new_out = np.array([])
            for tmp in out:
                if tmp <= 0.33:
                    new_out = np.append(new_out, 1)
                # elif tmp <= 0.66:
                #     new_out = np.append(new_out, 2)
                else:
                    new_out = np.append(new_out, 3)
            # out = [1. if tmp > 0.5 else 0 for tmp in out]
            target = np.where(target==1, 2, target)
            target = np.where(target==0.5, 1, target)
            target = np.where(target==0, 0, target)
            # print(new_out)
            # print(target)
            print(f'accuracy_score:{accuracy_score(new_out, target)}')
            # print(f'accuracy_score:{clf.score(new_out, target)}')
            print("i {}, valid {}".format(i, valid_loss.item()))
            print("         ")
```

```
            print( _________ )

        model.train()

    if i % 50 == 0 and best_loss > valid_loss.item():
        best_loss = valid_loss.item()
        cnt = 0
    else:
        cnt += 1

    if cnt > patience:
        break
print('training finished')

```

```
 31%|███▏      | 31/100 [00:14<00:30,  2.25it/s]batch 64 device cpu
 32%|███▏      | 32/100 [00:14<00:30,  2.26it/s]batch 64 device cpu
 33%|███▎      | 33/100 [00:14<00:29,  2.27it/s]batch 64 device cpu
 34%|███▍      | 34/100 [00:15<00:29,  2.24it/s]batch 64 device cpu
 35%|███▌      | 35/100 [00:15<00:29,  2.23it/s]batch 64 device cpu
 36%|███▌      | 36/100 [00:16<00:28,  2.24it/s]batch 64 device cpu
 37%|███▋      | 37/100 [00:16<00:28,  2.22it/s]batch 64 device cpu
 38%|███▊      | 38/100 [00:17<00:27,  2.24it/s]batch 64 device cpu
 39%|███▉      | 39/100 [00:17<00:27,  2.24it/s]batch 64 device cpu
 40%|████      | 40/100 [00:18<00:26,  2.25it/s]batch 64 device cpu
 41%|████      | 41/100 [00:18<00:26,  2.25it/s]batch 64 device cpu
 42%|████▏     | 42/100 [00:18<00:25,  2.26it/s]batch 64 device cpu
 43%|████▎     | 43/100 [00:19<00:25,  2.27it/s]batch 64 device cpu
 44%|████▍     | 44/100 [00:19<00:24,  2.27it/s]batch 64 device cpu
 45%|████▌     | 45/100 [00:20<00:24,  2.27it/s]batch 64 device cpu
 46%|████▌     | 46/100 [00:20<00:23,  2.25it/s]batch 64 device cpu
 47%|████▋     | 47/100 [00:21<00:23,  2.26it/s]batch 64 device cpu
 48%|████▊     | 48/100 [00:21<00:23,  2.26it/s]batch 64 device cpu
 49%|████▉     | 49/100 [00:22<00:22,  2.22it/s]batch 64 device cpu
 50%|█████     | 50/100 [00:22<00:22,  2.24it/s]batch 64 device cpu
 51%|█████     | 51/100 [00:23<00:26,  1.88it/s]accuracy_score:0.296875
i 50, valid 0.9732868671417236

_________
batch 64 device cpu
 52%|█████▏    | 52/100 [00:23<00:24,  1.98it/s]batch 64 device cpu
 53%|█████▎    | 53/100 [00:24<00:22,  2.07it/s]batch 64 device cpu
```

```
54%|█████▍    | 54/100 [00:24<00:21,  2.12it/s]batch 64 device cpu
55%|█████▌    | 55/100 [00:24<00:20,  2.16it/s]batch 64 device cpu
56%|█████▌    | 56/100 [00:25<00:20,  2.18it/s]batch 64 device cpu
57%|█████▋    | 57/100 [00:25<00:20,  2.14it/s]batch 64 device cpu
58%|█████▊    | 58/100 [00:26<00:19,  2.16it/s]batch 64 device cpu
59%|█████▉    | 59/100 [00:26<00:18,  2.17it/s]batch 64 device cpu
60%|██████    | 60/100 [00:27<00:18,  2.19it/s]batch 64 device cpu
61%|██████    | 61/100 [00:27<00:17,  2.19it/s]batch 64 device cpu
62%|██████▏   | 62/100 [00:28<00:17,  2.20it/s]batch 64 device cpu
63%|██████▎   | 63/100 [00:28<00:16,  2.20it/s]batch 64 device cpu
64%|██████▍   | 64/100 [00:29<00:16,  2.22it/s]batch 64 device cpu
65%|██████▌   | 65/100 [00:29<00:15,  2.20it/s]batch 64 device cpu
66%|██████▌   | 66/100 [00:29<00:15,  2.21it/s]batch 64 device cpu
67%|██████▋   | 67/100 [00:30<00:14,  2.21it/s]batch 64 device cpu
68%|██████▊   | 68/100 [00:30<00:14,  2.19it/s]batch 64 device cpu
69%|██████▉   | 69/100 [00:31<00:14,  2.21it/s]batch 64 device cpu
70%|███████   | 70/100 [00:31<00:13,  2.22it/s]batch 64 device cpu
71%|███████   | 71/100 [00:32<00:12,  2.25it/s]batch 64 device cpu
72%|███████▏  | 72/100 [00:32<00:12,  2.24it/s]batch 64 device cpu
73%|███████▎  | 73/100 [00:33<00:12,  2.24it/s]batch 64 device cpu
74%|███████▍  | 74/100 [00:33<00:11,  2.24it/s]batch 64 device cpu
75%|███████▌  | 75/100 [00:33<00:11,  2.25it/s]batch 64 device cpu
76%|███████▌  | 76/100 [00:34<00:10,  2.26it/s]batch 64 device cpu
77%|███████▋  | 77/100 [00:34<00:10,  2.24it/s]batch 64 device cpu
78%|███████▊  | 78/100 [00:35<00:09,  2.26it/s]batch 64 device cpu
79%|███████▉  | 79/100 [00:35<00:09,  2.26it/s]batch 64 device cpu
80%|████████  | 80/100 [00:36<00:08,  2.26it/s]batch 64 device cpu
81%|████████  | 81/100 [00:36<00:08,  2.26it/s]batch 64 device cpu
82%|████████▏ | 82/100 [00:37<00:07,  2.28it/s]batch 64 device cpu
83%|████████▎ | 83/100 [00:37<00:07,  2.28it/s]batch 64 device cpu
84%|████████▍ | 84/100 [00:37<00:07,  2.27it/s]batch 64 device cpu
85%|████████▌ | 85/100 [00:38<00:06,  2.27it/s]batch 64 device cpu
86%|████████▌ | 86/100 [00:38<00:06,  2.25it/s]batch 64 device cpu
87%|████████▋ | 87/100 [00:39<00:05,  2.26it/s]batch 64 device cpu
```

```
1 model.eval()
2
3 def predict(wavfile):
4     waveform, _ = librosa.load(wavfile, sr=16000)
5
6     input = _timit_dataloader.preprocess_sample(waveform)
7     with torch.no_grad():
```

```
    with torch.no_grad():
        out = model(torch.tensor(input, dtype=torch.float).to(device))
        out = out.cpu().detach().numpy()
    print(out)
    if out <= 0.33:
        out = '<25'
    elif out <= 0.66:
        out = '>25<40'
    else:
        out = '>40'
    # out = 'female' if out < 0.5 else 'male'
    return out
```

```
# Code for recording audio from the browser
from IPython.display import Javascript
from google.colab import output
from base64 import b64decode
import IPython
import uuid
from google.colab import output


class InvokeButton(object):
  def __init__(self, title, callback):
    self._title = title
    self._callback = callback

  def _repr_html_(self):
    from google.colab import output
    callback_id = 'button-' + str(uuid.uuid4())
    output.register_callback(callback_id, self._callback)

    template = """<button id="{callback_id}" style="cursor:pointer;background-color:#EEEEEE;border-color:#E0E0E0;padding:5px 15px
        <script>
          document.querySelector("#{callback_id}").onclick = (e) => {{
            google.colab.kernel.invokeFunction('{callback_id}', [], {{}})
            e.preventDefault();
          }});
```

```
          });
        </script>"""
    html = template.format(title=self._title, callback_id=callback_id)
    return html

RECORD = """
const sleep  = time => new Promise(resolve => setTimeout(resolve, time))
const b2text = blob => new Promise(resolve => {
  const reader = new FileReader()
  reader.onloadend = e => resolve(e.srcElement.result)
  reader.readAsDataURL(blob)
})
var record = time => new Promise(async resolve => {
  stream = await navigator.mediaDevices.getUserMedia({ audio: true })
  recorder = new MediaRecorder(stream)
  chunks = []
  recorder.ondataavailable = e => chunks.push(e.data)
  recorder.start()
  await sleep(time)
  recorder.onstop = async ()=>{
    blob = new Blob(chunks)
    text = await b2text(blob)
    resolve(text)
  }
  recorder.stop()
})
"""

def record(sec=3):
  display(Javascript(RECORD))
  s = output.eval_js('record(%d)' % (sec*1000))
  b = b64decode(s.split(',')[1])
  with open('audio.wav','wb+') as f:
    f.write(b)
  return 'audio.wav'
```

```
def classify():
  print("Now recording for 3 seconds...say what you will...")
```

```
  print('Now recording for 3 seconds, say what you will...')
  record()
  os.system('ffmpeg -i audio.wav -ar 16000 -y audio.wav')
  print(f"Audio recording complete, guess it is {predict('audio.wav')}")

InvokeButton('Start recording', classify)
```

Start recording

```
Now recording for 3 seconds, say what you will...
[0.00109952]
Audio recording complete, guess it is <25
Now recording for 3 seconds, say what you will...
[7.352042e-05]
Audio recording complete, guess it is <25
Now recording for 3 seconds, say what you will...
[0.00268975]
Audio recording complete, guess it is <25
Now recording for 3 seconds, say what you will...
[0.03905485]
Audio recording complete, guess it is <25
Now recording for 3 seconds, say what you will...
[0.00256719]
Audio recording complete, guess it is <25
```

```
IPython.display.Audio('audio.wav')
```

0:03 / 0:03

```
1
```