Robotics And Machine vision
EC-422
PROJECT REPORT

Feature Learning from Spectrograms for Assessment of Personality Traits

Submitted by:
Arindam Jain
(2K16/EE/027)

Submitted to:
Prof. Rajeev Kapoor

Index

Abstract
1. Introduction
2. DataSet
3. Artificial Neural Networks
4. Support Vector Machine
5. Feature Learning Method
   5.1 Feature Extraction
   5.2 Dictionary Learning
   5.3 Classification
6. References

Abstract

Several methods have recently been proposed to analyze speech and automatically infer the personality of the speaker. These methods often rely on prosodic and other hand crafted speech processing features extracted with off-the-shelf toolboxes. To achieve high accuracy, numerous features are typically extracted using complex and highly parameterized algorithms. In this paper, a new method based on feature learning and spectrogram analysis is proposed to simplify the feature extraction process while maintaining a high level of accuracy. The proposed method learns a dictionary of discriminant features from patches extracted in the spectrogram representations of training speech segments. Each speech segment is then encoded using the dictionary, and the resulting feature set is used to perform classification of personality traits. Experiments indicate that the proposed method achieves state-of-the-art results with an important reduction in complexity when compared to the most recent reference methods. The number of features, and difficulties linked to the feature extraction process are greatly reduced as only one type of descriptors is used, for which the 7 parameters can be tuned automatically. In contrast, the simplest reference method uses 4 types of descriptors to which 6 functionals are applied, resulting in over 20 parameters to be tuned.

1. Introduction

In the literature, five personality traits (the *Big-Five*) corre- sponding to psychological phenomena are observable regardless of the situation and culture: openness, conscientiousness, extroversion, agreeableness and neuroticism

Neuroticism (N): People whose ratings generate a high score in neuroticism are presumed to be emotionally unstable and easily shocked or ashamed. They are easily overwhelmed by feelings or nervousness and are gener- ally not self-confident. On the contrary, people with low ratings are presumed to be calm and stable. They work well under pressure and are not easily agitated.

Extroversion (E): High scores in extroversion indicate a sociable, energetic, independent personality while intro- verted personalities are presumed to be rather conserva- tive, reserved and contemplating.

Openness (O): The scores of the openness factor estimate the degree to which a person considers new ideas and integrates new experiences in everyday life. High rated persons are presumed to be visionary, curious. They perceive what's happening in the surrounding and are open to venturesome experiments. On the opposite side, people with low scores are generally conservative. They prefer common-knowledge to avant-garde.

Agreeableness (A): High scores in agreeableness suggest that people are rather sympathetic. They trust other people and are being helpful. Non-agreeable personalities are presumed to be egocentric, competitive and distrustful.

Conscientiousness (C): People of high scores in the consci- entiousness factor are presumed to be accurate, careful, reliable and effectively planning while people of low scores are presumed to act carelessly, not thoughtfully and improperly

Detecting personality is one of the most important marketing strategies in today's world. You could personalize different things for an individual specifically to suit their interest. For this reason, we decided to do a project where we could detect a person's emotions just by their voice which will let us manage many AI related applications. Some examples could be including call centers to play music when one is angry on the call. Another could be a smart car slowing down when one is angry or fearful. As a result this type of application has much potential in the world that would benefit companies and also even safety to consumers.

2. DataSet

The dataset includes the following material:
1) audio files
2) raw personality questionnaires
3) personality assessments
4) metadata

1) Audio files

The dataset includes 640 audio clips of 10 seconds
Audio files format: .WAV files(16 bit, mono, 8 kHz)

2) Raw Personality Questionnaires

This dataset includes the raw questionnaires filled individually by 11 assessors.
#1. This person is reserved
#2. This person is generally trusting
#3. This person tends to be lazy
#4. This person is relaxed, handles stress well
#5. This person has few artistic interests
#6. This person is outgoing, sociable
#7. This person tends to find fault with others
#8. This person does a thorough job
#9. This person gets nervous easily
#10. This person has an active imagination

3) Personality assessments

The dataset includes the personality scores assigned individually
by the 11 assessors. The scores are obtained from the raw personality
questionnaires (see the paper above for further information).
The five scores corresponds to the following traits:
Extraversion
Agreeableness
Conscientiousness
Neuroticism
Openness

4) metadata

The dataset includes metadata for each audio sample (see the
paper agove for further information):
speaker ID (integer between 1 and 322)
gender ("M" for male and "F" for female)
status ("J" for journalist and "G" for non-journalist)

3. Artificial Neural Networks

Artificial Neural Networks are computing systems roughly modeled after the Neural Networks of the human brain. Though not as efficient, they perform in roughly similar ways. These systems, just like the brain, learn from what they experience. Artificial Neural Networks learn tasks by comparing samples, generally without specifically assigned goals.

For example, while learning image recognition, Neural Networks in training would learn to identify images containing dogs by examining sample images that have been tagged with “dog” or “no dog” labels and then use those results to locate and identify dogs in new images. These Neural Networks start from zero, with no data about dog characteristics, such as tails, ears, and fur. The systems develop their own understanding of relevant characteristics based on the learning material being processed.

An Artificial Neural Network uses a collection of connected nodes called artificial neurons – a simple imitation of the biological neurons. The connections are versions of synapses and operate when an artificial neuron transmits a signal from one to another. The artificial neuron that receives the signal can process it and then signal artificial neurons connected to it.

One significant advantage of Neural Networks is their ability to learn in nonlinear ways. This means they have the ability to spot features in an image that are not obvious. For example, when identifying oranges, Neural Networks could spot some in direct sunlight and others in the shade on a tree, or they might spot a bowl of oranges on a shelf in a picture with a different subject. This ability is the result of an activation layer designed to highlight the useful details in the identification process.

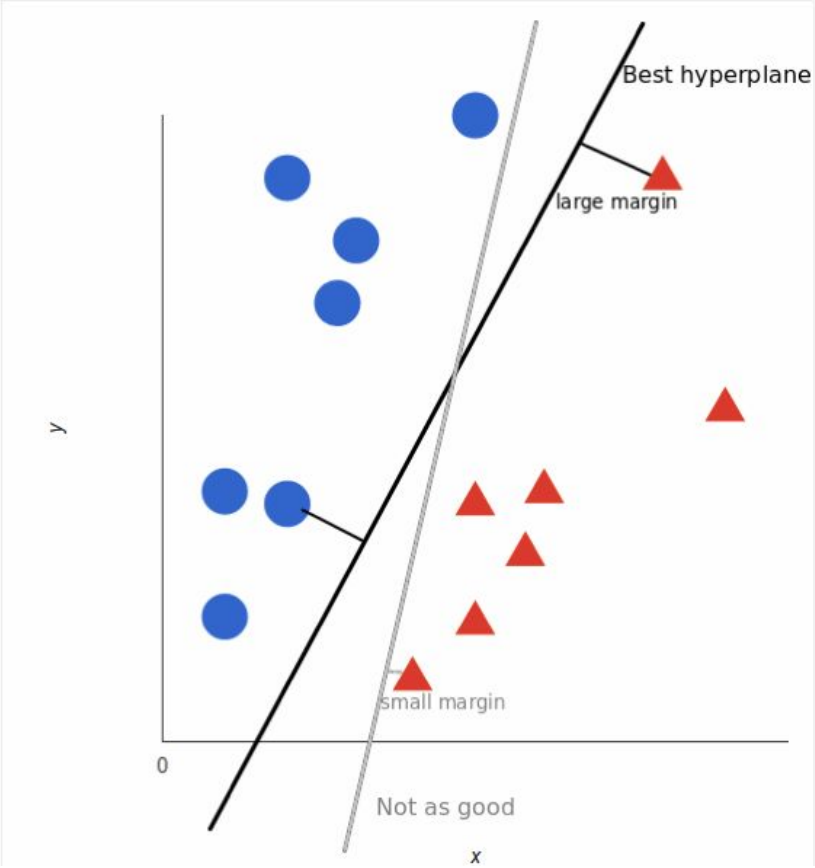
There are six types of Neural Networks, but two are the most popular: Recurrent and Feedforward. A Feedforward Neural Network sends data in one direction only. Data moves from input nodes, through hidden nodes (if any exist), and to the output nodes. Feedforward Neural Networks do not use loops or cycles and are considered the simplest type of Neural Network. This type of system can include many hidden layers.

4. Support vector machine (SVM)

A support vector machine (SVM) is a supervised machine learning model that uses classification algorithms for two-group classification problems. After giving an SVM model sets of labeled training data for each category, they're able to categorize new text.

How Does SVM Work?

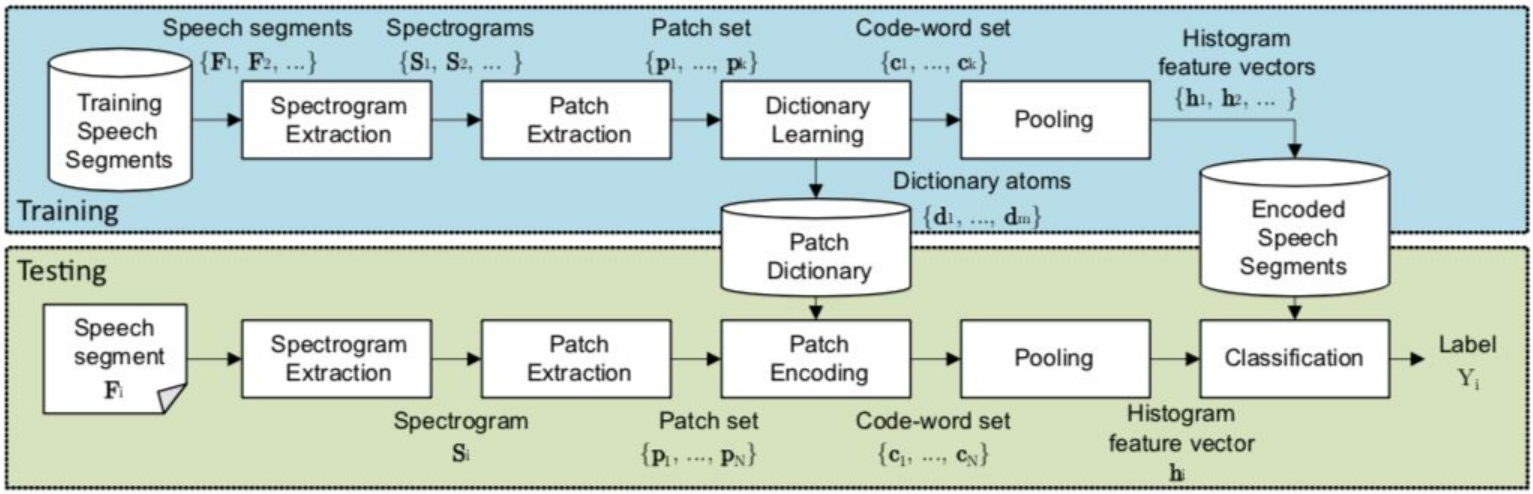
The basics of Support Vector Machines and how it works are best understood with a simple example. Let's imagine we have two tags: *red* and *blue*, and our data has two features: *x* and *y*. We want a classifier that, given a pair of *(x,y)* coordinates, outputs if it's either *red* or *blue*. We plot our already labeled training data on a plane:



5. FEATURE LEARNING METHOD

This section presents a new method for predicting personal- ity traits in speech based on spectrogram analysis and feature learning. The main stages of the proposed method are depicted in Figure 1. Specific details regarding our proposed solution for feature extraction, classification and dictionary learning are described in the next sections. The upper part is the pipeline for training. At first, for each speech segment F in the data set, a spectrogram S is extracted by applying a Fourier transform on a sliding window, yielding a 2-dimensional matrix. Small sub-matrices, called patches {p1 , ..., pk } are then uniformly extracted from all the spectrogram matrices in the training set. A dictionary D = {d1 , ..., dm } is learned from these patches, and at the same time, the patches are encoded as sparse vectors called code-words {c1 , ..., ck }. A single m- dimensional feature vector representation h is obtained for each training speech sample by pooling together all code- words extracted from it. A two-class support vector machine (SVM) classifier is trained using these feature vectors for each personality trait.

The lower part is the pipeline used during testing, to predict a personality trait. Like in training, patches are extracted from the spectrograms. Each patch is encoded using the previously learned dictionary. The resulting code-words are then pooled to create a summarizing feature vector. This vector is used by a 2-class classifier which predicts if the speaker exhibits, or not, a specific personality trait.



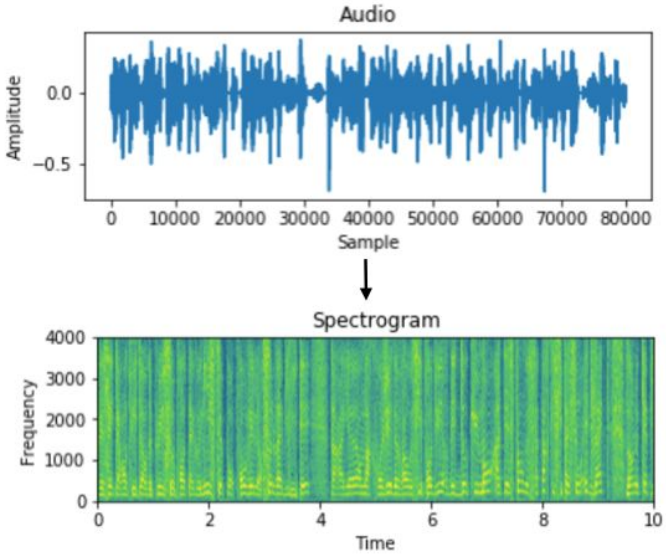
5.1 **Feature Extraction**

Given a speech segment x(n), the spectrogram S is the concatenation in time of its windowed DFT:

S = {X0, ..., Xt, ..., XT }

where Xt is a column vector containing the absolute amplitude of the DFT frequency bins and T is the number of DFTs extracted from the signal. The absolute amplitude is favored over the log- amplitude as it has shown to yield better results for spectrogram image classification in and in our own experiments. The spectrograms are normalized: each frequency bin is divided by the maximum amplitude value contained in a time frame. This process results in a 2-D matrix S which can be analyzed as a grey-scale image. An example of spectrogram extracted on the SSPNet Speaker Personality Corpus is illustrated in Figure 2.

From the matrix S, small patches, or sub-images, of p × p pixels are extracted at regular intervals. A vector representation pi ∈ R1×d of each patch (d = p × p) is obtained by concatenating the value of all pixels. The vector pi is encoded into ci using a previously learned dictionary D containing m atoms. These atoms are vector basis that are used to reconstruct the patches. The code-vector ci corresponding to the patch pi is obtained by solving using the LARS-Lasso algorithm. The loss function has two terms, each encoding an optimiza- tion objective, and λ is a parameter used to adjust the relative importance of the two terms. The first term is the quadratic reconstruction error, while in the second term, the l1 norm of the code vector is used to enforce sparseness. Once a code ci is obtained for each patch pi, the absolute value of all the codes are summed to obtain a histogram h describing the entire spectrogram. These histograms represent the distribution of patches over speech segments. It is thus possible to directly compare segments of different length.



5.2 **Dictionary Learning**

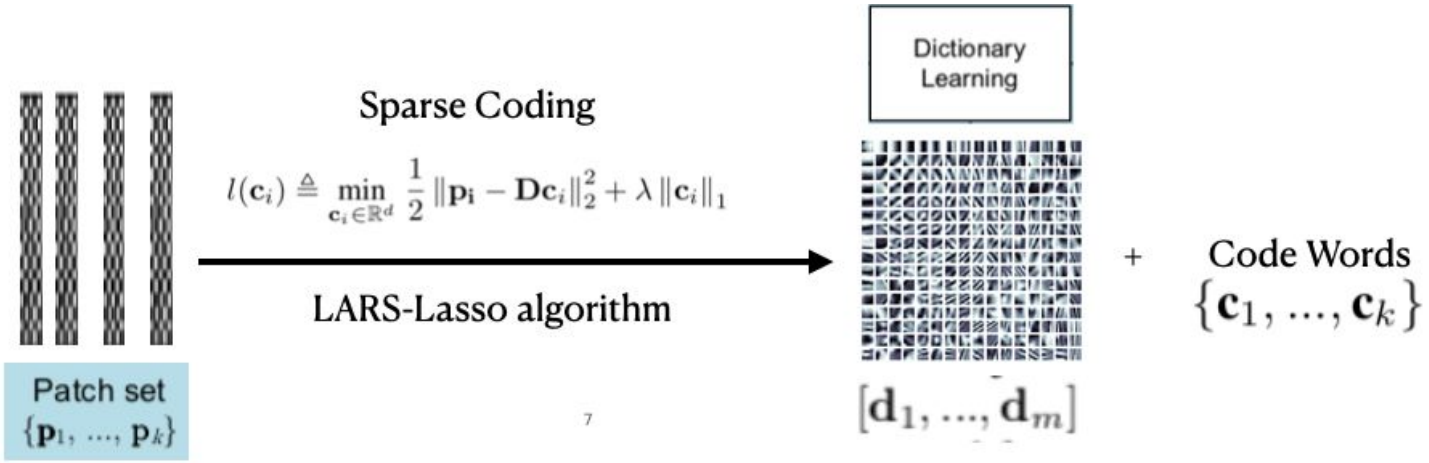
The objective of the dictionary learning phase is to generate a representative dictionary D = [d1,...,dm] ∈ Rd×m given the matrix P = [p1,...,pk] ∈ Rd×k containing patch vectors ex- tracted from the training set. Generally, for image classification tasks, best results are obtained with over-complete (m > d) dictionaries [55].

A dictionary of atoms D and sparse code-words C can be obtained by minimizing the following loss function:

$$l(\mathbf{c}_i) \triangleq \min_{\mathbf{c}_i \in \mathbb{R}^d} \frac{1}{2} \|\mathbf{p_i} - \mathbf{D}\mathbf{c}_i\|_2^2 + \lambda \|\mathbf{c}_i\|_1$$

In this equation, λ is the same as in (2) and is used to adjust the weight of the sparseness term in the loss equation.

The first is used to restrict the magnitude of the dictionary atoms. The second is used to make sure each element of each atom in the dictionary is positive. Since the spectrogram is purely positive, better results are obtained by enforcing this constraint. The joint optimization of C and D is not convex. However if one term is fixed the problem becomes convex. Thus, a common strategy is two alternates between updating C while D is fixed and updating D while C is fixed until a stopping criterion is met.



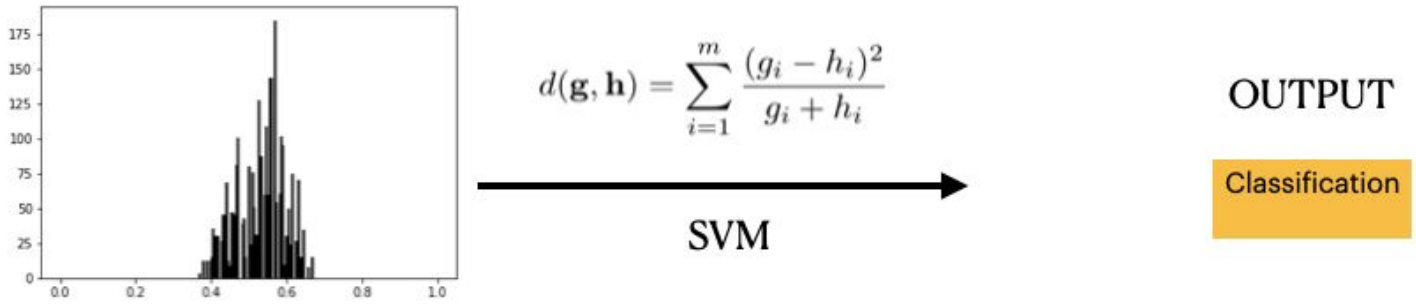
5.3 **Classification**

The speech segments are represented by histograms and thus, appropriate distance measure should be employed. Several distance measures have been proposed to compare histograms. In this paper's implementation, the χ2 distance is used because it showed competitive performances for visual bag- of-words histograms. The χ2 distance is given by :

$$d(\mathbf{g}, \mathbf{h}) = \sum_{i=1}^{m} \frac{(g_i - h_i)^2}{g_i + h_i}$$

where gi and hi are the ith bins of histograms h and y, and m corresponds to the number of words

While the implementation of this paper employs the χ2 distance and an SVM classifier, the proposed methods is not bound to these choices, and other distance functions and classifiers can be used.



6. **Implementation Results and conclusion**

Performance of the proposed and baseline methods on the SSPNet Speaker Personality corpus is reported in Table I. The best average UAR was obtained using the proposed method. However, the results obtained when using the challenge fea- tures and GeMAPS with an SVM classifier are comparable. The method proposed by Mohammadi and Vinciarelli yields slightly lower accuracy than the other methods, although the difference in performance in most cases is small and may be negligible. Particularities in the data set and the type of classifier, as well as its implementation, are most likely the reason for these variations in performance.

Algorithm	Kernel	gamma	degree	Accuracy
SVM	rbf	0.01	2	64.3%
SVM	polynomial	0.001	3	61.7%
SVM	linear	0.1	1	58.2%

This paper presents a new method for automated assessment of personality traits in speech. Speech segments are represented using spectrograms and feature learning. The proposed representation is compact and is obtained using a single algorithm requiring minimal expert intervention, when com- pared to reference methods. Experiments conducted on SSPNet data set indicate that the proposed method yields the same level of accuracy as state-of-the-art methods in paralinguistics that employ more complex representations, while remaining simpler to use.

Time Complexity = O(n^2) + O(n) = O(n^2)
[Patch Extraction] [Feature Extraction]

Implementation Code -

https://github.com/Arindam-Jain/personality-detection (link to code)

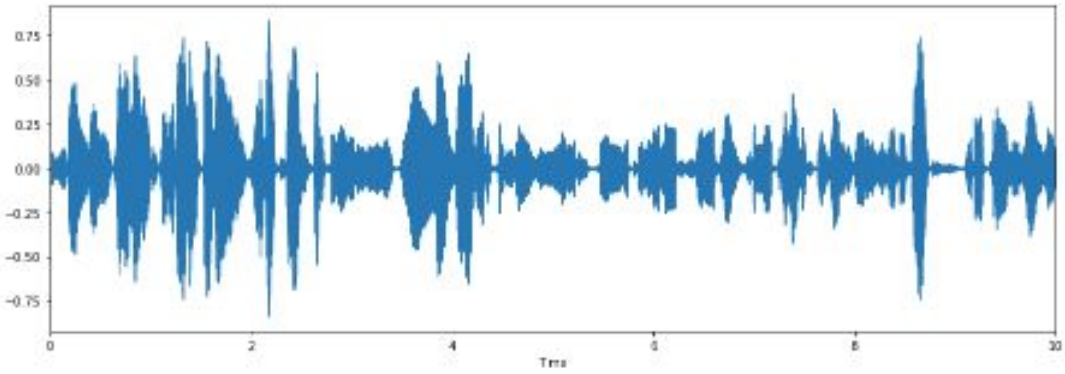
Plotting the audio file's waveform and its spectrogram

In [9]:
```
data, sampling_rate = librosa.load('SSPNet-Speaker-Personality-Corpus/Audio_clips/cut_feb
```

In [10]:
```
%% pylab inline
import os
import pandas as pd
import librosa
import glob

plt.figure(figsize=(15, 5))
librosa.display.waveplot(data, sr=sampling_rate)
```

Out[10]:
```
<matplotlib.collections.PolyCollection at 0x1307e0438>
```



In [11]:
```
IPython.display.Audio('SSPNet-Speaker-Personality-Corpus/Audio_clips/cut_feb0202-human-14
```

Out[11]:

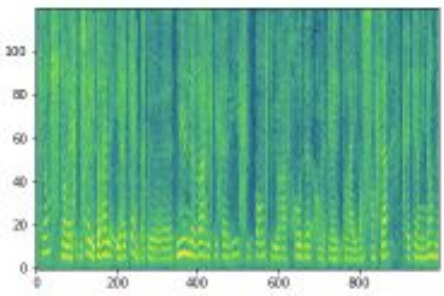
In [12]:
```
import matplotlib.pyplot as plt
import scipy.io.wavfile
import numpy as np
import sys


sr,x = scipy.io.wavfile.read('SSPNet-Speaker-Personality-Corpus/Audio_clips/cut_feb0202-h

for i,n in enumerate(nn):
    xseg = x[n-nwin:n]
    z = np.fft.fft(window * xseg, nfft)
    X[i,:] = np.log(np.abs(z[:nfft//2]))

plt.imshow(X.T, interpolation='nearest',
    origin='lower',
    aspect='auto')

plt.show()
```



Getting the features of audio files using librosa

In [20]:
```
f,l=feature()
```

```
Extraversion
(64,) (6,)
(64,) (6,)
(64,) (6,)
(64,) (6,)
(64,) (6,)
(64,) (6,)
(64,) (6,)
(64,) (6,)
(64,) (6,)
(64,) (6,)
(64,) (6,)
(64,) (6,)
(64,) (6,)
(64,) (6,)
(64,) (6,)
(64,) (6,)
(64,) (6,)
(64,) (6,)
(64,) (6,)
(64,) (6,)
```

In [21]:
```
np.save('feature_audio_new.npy', f)

np.save('label_audio_new.npy', l)
```

f = np.load('feature_audio_1.npy') l = np.load('label_audio_1.npy')

In [22]:
```
f[:2]
```

Out[22]:
```
array([[-2.73165725e+02,  1.68506560e+02, -8.26776758e+01,
         4.52161458e+01, -2.73800063e+00, -3.63990036e+01,
         1.10157531e+01, -3.05618961e+01, -5.86670067e+00,
         7.98776183e-01, -2.81447085e+01,  4.03014018e+00,
        -6.74525415e+00, -1.06286906e+01,  9.26697797e+00,
        -1.07126985e+01, -3.15052212e+00,  1.89655348e+00,
        -1.16933006e+01, -2.82994914e+00, -6.99870917e+00,
        -7.99619714e+00, -2.42582561e+00, -7.73633228e+00,
        -1.94866823e+00, -3.35481630e+00, -5.29155970e+00,
         2.76590459e+00, -1.08361880e+00, -6.28858914e-01,
         2.91566266e+00,  3.24228242e-01,  2.36977995e+00,
         1.91938894e+00,  3.03247735e+00,  5.64422385e+00,
         3.73470124e+00,  5.34778301e+00,  4.57695733e+00,
         2.15218095e+00,  3.12273184e+00,  1.63097593e+00,
         7.51506296e-01, -1.70366015e-01, -1.04416384e+00,
         7.00913845e-01, -1.65450435e-01, -1.19973529e+00,
        -9.09971657e-01, -9.61602811e-01, -9.52792044e-02,
        -8.41778000e-01, -1.08433587e+00, -3.25478136e-01,
        -1.13956429e+00, -1.04096956e+00, -1.02528297e+00,
        -8.79038361e-01, -1.03206776e-01, -8.18608289e-01,
        -5.86741881e-01, -4.44259590e-01, -1.42373443e+00,
```

In [23]:
```
import sklearn
from sklearn.svm import SVC
from sklearn.model_selection import train_test_split
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import classification_report



clf = SVC(kernel='rbf',C=5,gamma=0.01, degree=3,probability=True)
clf.fit(f, l.ravel())
```

Out[23]:
```
SVC(C=5, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape='ovr', degree=3, gamma=0.01, kernel='rbf',
  max_iter=-1, probability=True, random_state=None, shrinking=True,
  tol=0.001, verbose=False)
```

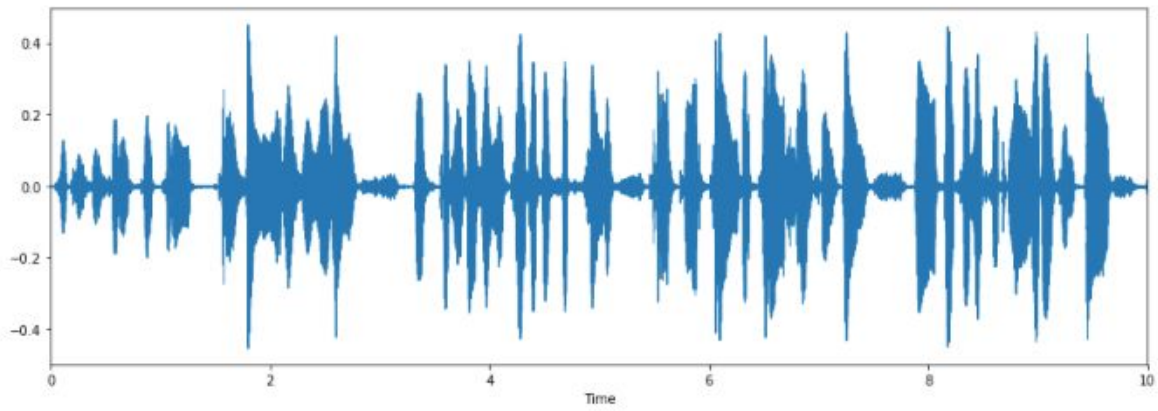
DEMO

In [42]:
```
file_name='SSPNet-Speaker-Personality-Corpus/Audio_clips/cut_feb0805-human-2.wav'
data, sampling_rate = librosa.load(file_name)
```

In [43]:
```
import os
import pandas as pd
import librosa
import glob

plt.figure(figsize=(15, 5))
librosa.display.waveplot(data, sr=sampling_rate)
```

Out[43]: <matplotlib.collections.PolyCollection at 0x1382f0048>



In [44]:
```
IPython.display.Audio(file_name)
```

Out[44]:

In [45]:
```
mfccs,tonnetz=parser(file_name)
print(mfccs.shape,tonnetz.shape)
demo_features=np.empty((0,70))
demo_ext_features = np.hstack([mfccs,tonnetz])
demo_features = np.vstack([demo_features,demo_ext_features])
```

```
(64,) (6,)
```

In [46]:
```
clf.predict_proba(demo_features)
```

Out[46]:
```
array([[0.20227236, 0.19950592, 0.19885065, 0.19629039, 0.20308068]])
```

In [47]:
```
clf.predict(demo_features)
```

Out[47]:
```
array([4.])
```

Below is the prediction number relation

1 : Extraversion

2 : Agreeableness

3 : Conscientiousness

4 : Neuroticism

5 : Openness

Actual Paper Results

Experiments should also be conducted where the sparse dictionary learning and classifier algorithms used in our implementation is replaced by other methods enforcing group sparsity and discrimination. Finally, given the unsupervised nature of the feature learning process, experiments should be conducted to assess the potential benefits of using a larger number of examples from other speech data sets.

PERFORMANCE ON THE SSPNET SPEAKER PERSONALITY CORPUS AND PARAMETER COMPLEXITY OF THE METHODS.

	Unweighted Average Recall (%)						**Number of**			
Algorithm	O	C	E	A	N	Avr.	Features	Descriptors	Functionals	Parameters
Mohammadi & Vinciarelli (LR) [18]	56.1	69.6	72.4	55.7	67.4	64.2	24	4	6	>20
Mohammadi & Vinciarelli (SVM) [18]	57.7	68.0	74.3	57.4	65.5	64.6	24	4	6	>20
Interspeech Challenge Baseline (SVM) [8]	**58.7**	69.2	74.5	62.2	69.0	66.7	6125	21	39	>200
Interspeech Challenge Baseline (RF) [8]	52.9	69.0	**77.5**	60.1	68.2	65.5	6125	21	39	>200
GeMAPS (SVM) [15]	56.3	72.2	74.9	61.9	68.9	66.8	62	13	10	>100
eGeMAPS (SVM) [15]	53.7	**72.5**	75.1	62.0	66.6	66.0	88	16	12	>100
SAE 1-Layer (SVM) [29]	57.1	64.3	69.2	62.0	65.8	63.7	100-800	1	1	>30
SAE 2-Layers (SVM) [29]	57.3	63.6	69.0	60.3	61.9	62.4	100-800	1	1	>30
Proposed Method	56.3	68.3	75.2	**64.9**	**70.8**	**67.1**	200-800	1	1	7

Time Complexity in Paper

The complexity of the LARS-lasso algorithm grows linearly on the size of the dictionary (O(m))

Time complexity of the proposed method also grows linearly on the length of the speech segment (O(n))

Finally, one could argue that more memory is required by the proposed method to store the dictionary (O(md))

2. References

[1] J. S. Uleman, L. S. Newman, and G. B. Moskowitz, “People as flexible interpreters: Evidence and issues from spontaneous trait inference,” *Adv. Exp. Soc. Psy.*, vol. 28, pp. 211–280, 1996.

[2] A. Tapus and M. J. Mataric, “Socially Assistive Robots: The Link between Personality, Empa- thy, Physiological Signals, and Task Performance.” in *Assoc. for the Adv. of Artif. Int. Spring Symp.*, 2008.

[3] J. M. Digman, “The curious history of the five-factor model.” *The Five-Factor Model of Per- sonality*, p. 20, 1996.

[4] R.E.Guadagno,B.M.Okdie,andC.A.Eno,“Whoblogs?Personalitypredictorsofblogging,” *Computers in Human Behavior*, vol. 24, no. 5, pp. 1993–2004, Sep. 2008.

[5] S. Argamon, S. Dhawle, M. Koppel, and J. W. Pennebaker, “Lexical predictors of personality type,” in *Joint Annu. Meeting of the Interface and the Classfication Soc. of North America*, 2005.

[6] F. Mairesse, M. A. Walker, M. R. Mehl, and R. K. Moore, “Using Linguistic Cues for the Automatic Recognition of Personality in Conversation and Text,” *J. Artif. Int. Res.*, vol. 30, no. 1, pp. 457–500, Nov. 2007.

[7] L. Qiu, H. Lin, J. Ramsay, and F. Yang, “You are what you tweet: Personality expression and perception on Twitter,” *J. of Res. in Personality*, vol. 46, no. 6, pp. 710–718, Dec. 2012.

[8] A. Vinciarelli, F. Burkhardt, R. V. Son, F. Weninger, F. Eyben, T. Bocklet, G. Mohammadi, and B. Weiss, “The Interspeech 2012 Speaker Trait Challenge,” in *Proc. Annu. Conf. of the Int. Speech Commun. Assoc.*, Portland, USA, 2012.

[9] C. Chastagnol and L. Devillers, “Personality Traits Detection Using a Parallelized Modified SFFS Algorithm,” in *Proc. Annu. Conf. of the Int. Speech Commun. Assoc.*, Portland, USA, 2012.

[10] Feature Learning from Spectrograms for Assessment of Personality Traits, Marc-Andre´ Carbonneau∗, Eric Granger, Yazid Attabi & Ghyslain Gagnon