Personality traits detection using a parallelized modified SFFS algorithm

Clément Chastagno1,2, Laurence Devillers1,3

1Department of Human-Machine Interaction, LIMSI-CNRS, France
2University of Orsay PXXI, France
3University of Sorbonne PIV, GEMASS-CNRS, France

{cchastag,devil}@limsi.fr

Abstract

We present in this paper a contribution to the INTERSPEECH 2012 Speaker Trait Challenge. We participated in the Personality Sub-Challenge, where the main characteristics of speakers according to the five OCEAN dimensions had to be determined based on short audio recordings solely. We considered the task as a general optimization problem and applied a modified version of the SFFS algorithm, wrapped around a SVM classifier, along with parallelized parameter tuning. Our system has yielded higher scores on all five dimensions for the development set, adding almost 14 percentage points to the recognition score of the Openness dimension.

Index Terms: personality detection, feature selection

1. Introduction

Automatic personality detection aims at building systems that can understand the expression of a behavior conveyed by a person, be it vocally, facially, or by gestures. This problem is part of a more general domain called Affective Computing [1] and formalized in 1995. An active community has emerged on these topics, especially around the problem of emotion and affect recognition in voice. Although works on very precise aspects such as the variations in pronunciation of a given vowel under different expressed emotions to try to model the statistics of spectral features have been conducted [2], data mining and statistical learning techniques are more prevalent in the domain. This approach requires both good training data and robust learning algorithm, so the community is increasingly pushing to collect naturally occurring emotional data [3] and applying sophisticated machine learning techniques to optimize recognition performance [4]. From this perspective, competitions have been organized to stimulate the community and disseminate corpora and good practices [5][6].

While efforts were initially focused on simple data with recordings of exaggerated expressions displayed by a small number of actors [7], they were then shifted to more spontaneous data, elicited in various simulated contexts such as a job interview [8] or a user test of a purposely faulty computer program [9]. The number of speakers in the gathered databases also increased, with the objective of training classifiers with a good enough power of generalization. The type of affective states investigated has also become more varied: first experiments usually reported on the detection of a subset of Ekman's basic emotions [10] but the community is now turning to more complex emotions and even more general speaker states and traits, e.g. age, gender [5] but also intoxication, sleepiness, interest... [6]. The INTERSPEECH 2012 Speaker Trait Challenge goes even further by proposing three new problems: detecting the personality, likability and pathology of a person using only short samples of his/her voice.

In the emotion detection field, as in many others, the data (here perceptively annotated audio segments) is usually represented numerically using thousands of features, often for a comparatively small number of instances. This is explained by the fact that, despite having good hints at emotion-related type of features (for instances, MFCC), no definitive feature set has been proven to perform sufficiently well for a wide range of databases and applications. Inspired by generative approaches [11], it is thus common to compute a large number of features by applying functionals to low-level descriptors in a combinatorial way [12]. This can yield better features indeed, especially when working with a very particular recognition task or database with no prior knowledge. Unfortunately, although adding dimensions does not increase misclassification probability when working with the entire population [13], it always does in practice because only a finite number of instances is available. With such high-dimensional data, not all dimensions are useful regarding the recognition task. More robust, lightweight and even better performing models can be trained when using only certain parameters, sometimes very few in comparison to the original data representation. Hence dimension reduction techniques have to be applied to extract low-dimensional structures on which robust classifiers can be trained.

Feature selection is a dimensionality reduction technique that has received a lot of attention [14]. Two types of feature selection methods can be roughly distinguished. Feature ranking methods are fast because they only require the computation of a metric for each feature, then allowing for their ranking, but they do not account for intra-feature relations (e.g. XOR problem). Wrapper methods use the output of a classifier to explore a tree made of possible subsets of the complete feature set; they thus perform far better but at an enormous computational expense. Furthermore, no comprehensive search can be achieved because of the combinatorial nature of the problem (a feature set of size N can be partitioned into 2^N subsets which should all be tested) so heuristics have to be applied.

Feature selection techniques have already been applied successfully in a wide range of domains including affective computing [15][16].

In this paper we present a system for the 2012 edition of the INTERSPEECH Speaker Trait Challenge. We participated in the Personality Sub-Challenge, where the main characteristics of speakers according to the five OCEAN dimensions [17] had to be determined based on short audio recordings solely. We considered the task as a general optimization problem and applied a modified version of the SFFS algorithm, wrapped around a SVM classifier, along with parallelized parameter...
tuning. After a short presentation of the data in Section 2, we
present the modifications made to the classical SFFS algorithm in
Section 3. Results are then presented and as well as a
discussion on the performances in Section 4. The conclusions are
given in Section 5.

2. Dataset

We worked on the data provided by the organizing committee of
the challenge. It consists of 640 audio clips of about 10 seconds
each. The segments are divided into three sets: one for training,
one for development and one for testing, each representing
respectively 40%, 29% and 31% of the whole corpus. We used
only the training and development sets, the testing set being
stripped of labels and suitable only for final evaluation. The
audio segments are not separated by speaker within each set,
although there is only two segments per speaker on average on
the whole corpus.

3. Followed approach

Our feature selection algorithm is based on the classical
Sequential Floating Forward Search algorithm [18], a wrapper
method. SFFS typically alternates between a Sequential Forward
Search (SFS) where it tries to add a feature to an existing well-
performing feature subset (exploitation phase) and a Sequential
Backward Search (SBS) where it tries to remove a feature from a
subset of size k + 1 and compares to the score attained by the
best subset of size k (exploration phase). In its original flavor,
SFFS tries to add or remove all possible features at each iteration
i.e. for D features, SFS on a subset of size k tries D - k features;
SBS tries k features. A greedy version of SFFS stops each
iteration as soon as a positive gain of performance is found, thus
speeding the whole process a lot. We used a heuristic ordering
the sets to try at each iteration based on similarity with
previously computed iterations. This modification of SFFS,
called SFFSS-SSH (Set-Similarity Heuristic) [19], proved to
accelerate significantly the obtention of good feature subsets
compared to the classical and the greedy versions of SFFS.

The algorithm is initialized by ranking the individual
features. This is done by training a classifier and computing a
recognition score of each individual feature. The SFS phase is
then triggered with the best individual feature i.e. the algorithm
tries to add the D - k possible features at each iteration
so that the best performing feature subset is selected at each step.
Obviously there is no need to go through the SBS phase when such a set has been
found, because all the individual features have already been
evaluated in the initialization phase.

Let us now assume that the algorithm is in the SFS phase
with a current feature set X of size k. The significance, or gain in
classification performance achievable by adding feature x to the
set X is denoted as S(x,X). The idea is to estimate S(x,X) using
similar cases in the tested feature sets history, i.e. when x was
added to a set X* considered the most similar to X. The estimate
of S(x,X) is denoted as S'(x,X). The metric used to compute a
similarity score between two feature sets is the Jaccard index:

\[
S'(x, X) = E(\tilde{X} \cup x) - E(\tilde{X})
\]

where \( J(X,Y) \) is the Jaccard index, \( E(X) \) represents the
classification performance using the feature set X and in (2), Y
is picked in the history of previously computed sets which added
the feature k. For instance, suppose the current feature set X is
(24, 13, 32, 48) and we are evaluating the significance of feature
#7. Among all the previously evaluated feature sets, we have
records for the significance (in percentage points) of feature #7
for three sets:

- \( Y_1: S(7, (24, 9)) = 3.2 \)
- \( Y_2: S(7, (32, 13, 92)) = -0.4 \)
- \( Y_3: S(7, (23, 13, 32, 53)) = 2.9 \)

The Jaccard index for X and \( Y_1, Y_2, \) and \( Y_3 \) is respectively
0.2, 0.4 and 0.6. So we estimate the significance of adding
feature #7 to the feature set X by 2.9.

The \( D - k \) possible features are ordered with decreasing
estimated significance to try the most promising ones first. In the
SBS phase, parents are ordered in increasing order of estimated
significance \( S' \), so that we first try to remove the least significant
features.

<table>
<thead>
<tr>
<th>Index of family</th>
<th>Number of features in family</th>
<th>Family content</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>58</td>
<td>SumofRASTA-style filtered auditory spectrum</td>
</tr>
<tr>
<td>2</td>
<td>58</td>
<td>Sumofauditory spectrum (loudness)</td>
</tr>
<tr>
<td>3-28</td>
<td>58</td>
<td>RASTA-style auditory spectrum bands 1-26 (0–8 kHz)</td>
</tr>
<tr>
<td>29</td>
<td>56</td>
<td>Pitch (by Sub-Harmonic-Summation algorithm)</td>
</tr>
<tr>
<td>30</td>
<td>56</td>
<td>Jitter of Jitter</td>
</tr>
<tr>
<td>31</td>
<td>56</td>
<td>Jitter (Pitch Variations from voiced frame to voiced frame)</td>
</tr>
<tr>
<td>32</td>
<td>56</td>
<td>Logarithmic Harmonics to Noise Ratio</td>
</tr>
<tr>
<td>33-46</td>
<td>58</td>
<td>MFCC 1–14</td>
</tr>
<tr>
<td>47-60</td>
<td>58</td>
<td>Spectral characteristics (derived from the linear magnitude spectrum)</td>
</tr>
<tr>
<td>61</td>
<td>58</td>
<td>RMS Energy</td>
</tr>
<tr>
<td>62</td>
<td>58</td>
<td>Zero-Crossing Rate</td>
</tr>
<tr>
<td>63</td>
<td>56</td>
<td>Shimmer (Amplitude variations from voiced frame to voiced frame)</td>
</tr>
<tr>
<td>64</td>
<td>56</td>
<td>Probability of voicing</td>
</tr>
<tr>
<td>65-128</td>
<td>38</td>
<td>First temporal derivative for family #1-64</td>
</tr>
<tr>
<td>129</td>
<td>5</td>
<td>Pitch (functionals computed only from voiced segments)</td>
</tr>
</tbody>
</table>

The SFFSS-SSH algorithm described above is a sequential
algorithm, comparable to the exploration of a tree where the
nodes are the \( 2^n \) possible feature sets. But for each node, an
evaluation is performed: it consists in a classifier training. We
used an SVM classifier with a RBF kernel, provided by the
library LIBSVM [20]. The choice of an RBF kernel was justified
by the fact that we would be treating relatively low-dimensional
problems in a context of feature selection. The linear kernel was
more adapted to compute the baseline scores on the whole
feature set (6125 features). The parameters C and gamma were
optimized at each step by doing a grid search, each couple of
parameters (C, \( \gamma \)) being evaluated by a 5-fold cross-validation on
the training set to avoid over-fitting. The best parameters were
then used to train a classifier on the whole training set and tested
The UAR score obtained was the score E(X) for the current feature set X. The grid search was parallelized to speed up the calculation, each full iteration taking between one and two minutes depending on cluster load.

The number of original features was too large to run our algorithm in a reasonable amount of time so we grouped the features into families, a family corresponding to a vector of functionals being applied to the same LLD and not the other way round, because it would have little physical meaning.

We were thus selecting group of features rather than features; for instance, adding a family to an already existing feature set would effectively add between 5 and 58 features depending on the chosen family. This difference is explained by the fact that different functionals were applied to the different LLD. We were left with 129 families named according to the openSMILE specifications [21]. The granularity of the “family” definition is intentionally quite small to allow for future analysis of the selected families in relation to the investigated dimension. The details of the families can be found in Table 1 above.

To sum up the experimental setup, we fixed several parameters (number of cross-validation folds, definition of the families, choice of the wrapper classifier and of the SVM kernel) and let the parameters C and γ free for optimization. The algorithm was left running for a definite period of time (two days).

### 4. Results and discussion

The yielded results are presented in Table 2 above, comparing the baseline scores and ours. The number of families used along with the number of features (both in absolute and in percentage of the total feature set) are also presented. We gave the first five families selected by the algorithm. The results correspond to the smaller feature set with the maximum score (bigger sets with equal score were ignored). Overall all the scores are above the baseline for the development set, but this is not the case for the testing set. On the development set, gain from +7.9 percentage points (pp) on the Neuroticism dimension to +13.7pp on the Openness dimension are achieved. On the testing set, we have positive gains only on the Agreeableness and Openness dimensions (+4.8pp and +0.2pp respectively), while the worst performance occurs for the Conscientiousness dimension (-4.6pp). Yet, we see that we use far less features than the baseline system for all dimensions since only 8.7% to 16.6% of the 6125 original features were selected. The selected families are mainly energy and spectral related LLD, less often voice related LLD. For all dimensions, MFCC and RASTA-style auditory spectrum related families are among the five first selected families.

There is one obvious issue with our method: the generalization performance. The difference between the performance on the testing set and the development set is much lower for the baseline system than for our system (a maximum loss of respectively 7.4pp, -10.9% relative and 18.4pp, -20.1% relative). We think the reliability of the performances obtained on the development set can be improved by increasing the number of cross-validation folds and better estimating the C and γ parameters for the final model. The number of cross-validation folds was set to 5 to accelerate the computation but as the folds are built randomly at execution time, the cross-validation performance can vary. It varies even more as the number of folds decreases. Another issue is that the choice of parameters C and γ for the final training is made upon the cross-validation performance after a grid search. The shape of the three-dimensional C-γ-score surface is quite characteristic, usually a plateau sometimes featuring a few bumps [22]. This phenomenon, combined with the variations in score due to the random cross-validation folds, means that we can run into stability problems when running the algorithm several times. This is a problem if we want to analyze the selected feature families with respect to dimensions, not so much in a competition context, where we are mainly interested into optimizing recognition performance. Yet we feel that the feature selection techniques in general and the tool that we developed and used in particular are promising to develop a list of “good” and “bad” features for a given task, a true holy grail for the community. The construction of a metric of similarity between corpora using results of feature selection is also an interesting possibility [23]. We think that more work is needed to address the described issues.

### Table 2: Comparison between baseline and achieved UAR scores

<table>
<thead>
<tr>
<th></th>
<th>Openness</th>
<th>Conscientiousness</th>
<th>Extraversion</th>
<th>Agreeableness</th>
<th>Neuroticism</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Baseline UAR</strong>&lt;br&gt;score (dev. set)</td>
<td>60.4%</td>
<td>74.5%</td>
<td>80.9%</td>
<td>67.6%</td>
<td>68.0%</td>
</tr>
<tr>
<td><strong>SFFS-SSH UAR</strong>&lt;br&gt;score (dev. set)</td>
<td>74.1%</td>
<td>84.0%</td>
<td>91.8%</td>
<td>78.0%</td>
<td>75.8%</td>
</tr>
<tr>
<td><strong>Baseline UAR</strong>&lt;br&gt;score (testing set)</td>
<td>57.8%</td>
<td>80.1%</td>
<td>76.2%</td>
<td>60.2%</td>
<td>65.9%</td>
</tr>
<tr>
<td><strong>SFFS-SSH UAR</strong>&lt;br&gt;score (dev. set)</td>
<td>58.0%</td>
<td>75.5%</td>
<td>73.4%</td>
<td>65.0%</td>
<td>65.2%</td>
</tr>
<tr>
<td># families</td>
<td>14</td>
<td>14</td>
<td>11</td>
<td>16</td>
<td>21</td>
</tr>
<tr>
<td># features</td>
<td>670</td>
<td>710</td>
<td>534</td>
<td>773</td>
<td>1014</td>
</tr>
<tr>
<td>% of features used</td>
<td>10.9%</td>
<td>11.6%</td>
<td>8.7%</td>
<td>12.6%</td>
<td>16.6%</td>
</tr>
<tr>
<td>selected families (first five)</td>
<td>Family #26 Family #65 Family #61 Family #49 Family #32 Family #32 Family #46 Family #46 Family #66 Family #47 Family #57 Family #125 Family #85 Family #74 Family #85 Family #13 Family #124 Family #45 Family #64 Family #29 Family #93 Family #121 Family #99 Family #11 Family #92 Family #91</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
5. Conclusions

We presented a parallelized modified SFFS algorithm for our contribution to the INTERSPEECH 2012 Speaker Trait Challenge (Personality Sub-Challenge). We achieved better performance than baseline on the development set, with some strong improvements (almost 14 percentage points on the recognition of Openness), while using a fraction of the initial feature set (as low as 8.7%). The performance is less good on the testing set but our system still obtained an improvement of almost 5pp on the Agreeableness dimension and performed comparably to the baseline system for the Openness and Neuroticism dimensions. The heuristic modifying the classical SFFS algorithm and the parallelization of the classifier training ensured affordable computational time to use an otherwise prohibitive feature selection technique.

While the recognition scores are good, there are still issues to be addressed if we want this approach to play a significant role in more general problems as the construction of a list of “golden” features or the comparison of corpora using a feature selection based similarity metric.

Acknowledgments

This work is funded by the 2010-2013 French ANR ARMEN project (http://projet_armen.byethost4.com). The authors wish to thank Eric Bilinski for his invaluable help on using the computer cluster.

References