
Distinguishing Mislabeled Data from Correctly Labeled Data in Classifier Design

Sundara Venkataraman, Dimitris Metaxas,
Dmitriy Fradkin, Casimir Kulikowski,
Ilya Muchnik

DCS, Rutgers University, NJ

Overview

- Motivation
- Facial Expression Recognition Example
- Prior Work in Identifying Mislabeled Data
- A New Approach to Identifying Mislabeled Data
- Application to Facial Expression Recognition
- Experimental Results
- Discussion and Directions for Future Work

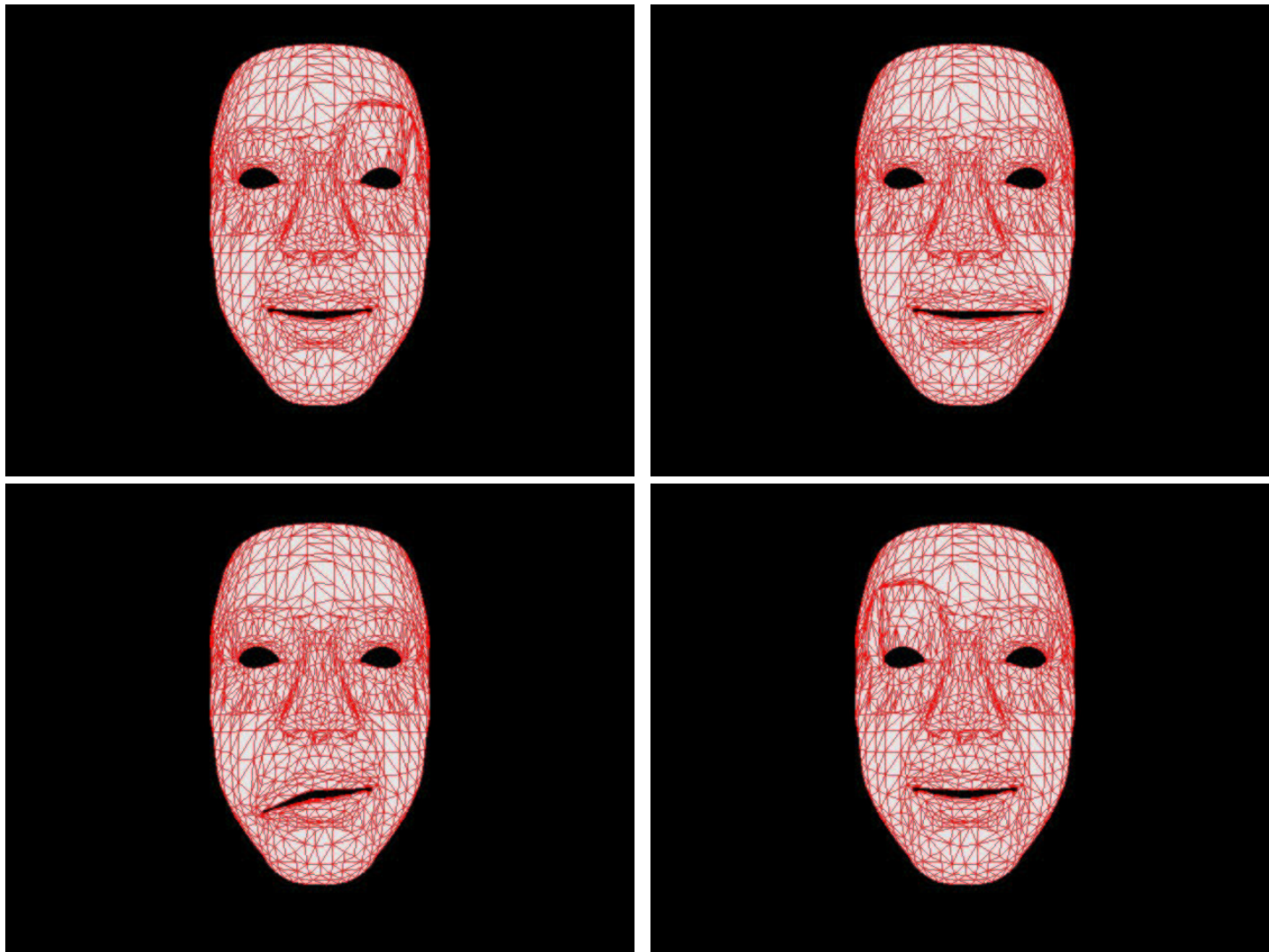
Motivation

- Frequently need to construct a classifier based on experimental data
- Experimental data may contain significant noise
- Experimental data may be incorrectly labeled
- These factors complicate the task of building and evaluating a reliable classifier

Example: Video Data Preparation

- Given a label (emotion) for a video sequence
- Select a short segment (2 seconds).
- Match a mask to the first frame (manually) and track it (automatically).
 - rigid alignment of the mask to the general direction
 - alignment of regions (mouth, eyebrows, etc.)
 - automatic fitting and tracking based on cue integration (edges, point tracker and optical flow) [Goldenstein, Vogler and Metaxas 2001, 2002, 2003]
- Obtain features for each frame: vector representation

Mask Deformations



Frame Features

1	Left eyebrow movement
2	Right eyebrow movement
3	Right lip stretch
4	Right lip curve
5	Left lip stretch
6	Left lip curve
7	Jaw movement
8	Scaling (rigid parameter)
9	Rotation about X axis
10	Rotation about Y axis
11	Rotation about Z axis
12	Translation about X axis
13	Translation about Y axis
14	Translation about Z axis

Table 1: Features of the frames.

Potential Problems

- The expression may not be present in every frame or time interval, so a bad segment may be selected
- The mask alignment or tracking may lead to errors in computing features
- Tracking errors can accumulate over time

We may end up with mislabeled data (i.e. data that doesn't correspond to the label assigned to it).

- There is a large literature on identifying outliers [Knorr and Ng 1997; Breunig et. al. 2000]. However, that is a somewhat different problem.
- Special methods for k Nearest Neighbors (kNN) [Sanchez et. al. 2003; Jiang and Zhou 2004].
- Removing “problem” points [Gamberger et. al 1999; Ganapathiraju and Picone 2000].
- A general approach based on using several independent classifiers [Brodley and Friedl 1999]. Applications and extentions to bagging and boosting in [Berthelsen and Megyesi 2000; Verbaeten and Assce 2003].

Our Approach

- A set of points is “good” if we can build a classifier with high cross-validation accuracy.
- Can’t exhaustively explore all possible subsets to find the best subset.
- Need to find a good set. (Then it may be extended by adding more points).
- Build classifiers in different (“discriminating”) subspaces and estimate their performance using cross-validation. This gives us statistics on probability of misclassifying a particular point.
- Consider the points that are misclassified by almost all classifiers “mislabeled”. Remove them from the set used to construct a classifier.

What Subspaces to Use?

- Representation may allow a natural partition into subspaces
- Use domain knowledge to find good subspaces.
- In a general case: SVMs with different kernel functions implicitly map points into different kernel spaces.

Application to Expression Recognition

- Given 2 videos of a person's facial expression (15-30 minutes).
- One is high stress expression, the other is low stress expression.
- Data available for 14 people (total of 28 sequences).

- Model-based approaches:
 - Each expression type is modeled by HMM [Cohen et. al 2003; Otsuka and Ohya, 1997].
 - Calculates likelihood function values for a given sequence of frames, measuring sequence's similarity to one of the predefined expression types
- Discriminative approaches:
 - Support Vector Machines (SVM) [Vapnik, 1998] applied to facial feature displacements [Michel and Kaliouby, 2003]

Our Sequence Representation

1. HMM constructed for each sequence S_i in the training set: Gaussian model used for the distribution of observations in each state.
2. Sequence represented by vector of parameters of HMM states, denoted by $r(S_i)$:

$$r_j(S_i) = \begin{cases} \mu_{ij}, & j = \overline{1, n} \\ v_i = [\mu_{i1}, \mu_{i2}, \dots, \mu_{in}], & j = 0 \end{cases} \quad (1)$$

μ_{ij} is the mean of the distribution of observations at state j of HMM H_i . (The $r_j(S_i)$ for $j = \overline{1, n}$ form a set of non-overlapping subspaces of representation $r_0(S_i) = v_i$).

3. So: all sequences are represented by vectors of the same dimensionality and standard classification methods, such as SVM, can be applied.

Experimental Pseudocode

Require: A set of k sequences S_i , $i = \overline{1, k}$, number of states $n > 1$.

- 1: **for** $i = 1, \dots, k$ **do**
- 2: Construct an HMM H_i with n states, based on sequence S_i .
- 3: Compute representative vector $r(S_i)$.
- 4: **end for**
- 5: Train SVM R on the set of vectors $r(S_i), i = 1, \dots, k$.
- 6: Output resulting classifier R .

Require: A sequence S_t , number of states $n > 1$.

- 1: Construct an HMM H_t with n states, based on sequence S_t
- 2: Compute representative vector $r(S_t)$.
- 3: Return the label given by classifier R to vector $r(S_t)$.

“Leave-one-person-out” cross-validation:

- Use two sequences of the same person as the test set
- Use the rest for training.
- Repeat this for each person.

Experimental Results I

Representation	$n = 2$		$n = 3$		$n = 4$	
v_i	18	64.3%	19	67.9%	18	64.3%
μ_{i1}	16	57.1%	17	60.7%	18	64.3%
μ_{i2}	15	53.6%	14	50.0%	17	60.7%
μ_{i3}			16	57.1%	17	60.7%
μ_{i4}					17	60.7%

Table 2: Results of cross-validation over all data (28 sequences).

Selecting Reliable Points

- Combine results of all experiments, i.e. over all values of n and choices of representation.
- For 7 out of 14 people one of their sequences would be misclassified at least 10 out of 12 times.
- Remove all sequences from these subjects.
- Retain the sequences from the other subjects as a “good” set.

Experimental Results II

Representation	$n = 2$		$n = 3$		$n = 4$	
v_i	11	78.6%	10	71.4%	11	78.6%
μ_{i1}	12	85.7%	12	85.7%	11	78.6%
μ_{i2}	13	92.9%	10	71.4%	10	71.4%
μ_{i3}			13	92.9%	12	85.7%
μ_{i4}					13	92.9%

Table 3: Results of cross-validation over “good” data (14 sequences).

Experimental Results III

Representation	$n = 2$		$n = 3$		$n = 4$	
v_i	8	57.1%	8	57.1%	8	57.1%
μ_{i1}	9	64.3%	9	64.3%	9	64.3%
μ_{i2}	7	50.0%	7	50.0%	7	50.0%
μ_{i3}			8	57.1%	6	42.9%
μ_{i4}					8	57.1%

Table 4: Results of cross-validation over unreliable data (14 sequences).

Results

- Overall the accuracy 16-18 out of 28 (57% – 64%): clearly better than random guessing, but not particularly good.
- On the selected “good” set the accuracy is 10-13 out of 14, or 71% - 93%.
- On the “bad” set the accuracy is 50% – 64%.

- The accuracy of classifiers trained on the “good” set when applied to the “bad” set were around 50%.
- The accuracy of classifiers trained on the “bad” set when applied to the good set were in the range 42%-57%.

Conclusion:

- The sequences in the “bad” set are not marginal points (otherwise a classifier built on them would generalize).
- Therefore they must be mislabeled.

Conclusion

- Suggested a general approach to extracting reliable data from sets of objects that are possibly mislabeled or incorrectly represented.
- Train the same classifier on different representations of the data, thought of as different subspaces of the same representation, to obtain statistics on labeling of each point.
- This distinguishes it existing methods that use different subsets of data to train classifiers or use different types of classifiers.
- The results of experiments with facial expression recognition data show that this approach is successful in finding a large subset of good objects.

A detailed study on benchmark datasets:

- To compare our approach with other existing methods
- To determine under what circumstances a particular method for finding mislabeled data should be used by itself or in conjunction with other methods.