Using Neural Networks to Include Semantic Information into Classification

Eduardo Ribeiro, Marcos Batista, Amar Daood, Eraldo Ribeiro, Celia Barcelos

1 Federal University of Goais, Brazil  
2 Florida Institute of Technology, U.S.A.  
3 University of Uberlandia, Brazil

Abstract

The accuracy of pattern-classification methods depends on how well the measured characteristics (i.e., features) represent the object to be classified. When using pre-designed features as it is the case of many pattern classifiers, one can try to enhance the features’ discriminative power by inserting high-level semantic information into the feature vectors. In this paper, we propose a method that increases the discriminative power of features by augmenting them with high-level semantic information learned from training data. Our method combines the advantages of dimensionality reduction techniques and feature-selection techniques. Instead of augmenting feature vectors, we map them using a modified neural network that has been trained to categorize the data into target groups. This neural network embeds categorization information. We tested the method on classification tasks for pollen species, human action, and acoustic signals. In all these tasks, our feature-enhancing method improved classification rates.

1 Introduction

Pattern classification is a classical problem in the fields of computer vision and pattern recognition. Its applications include classifying objects, scenes, and actions. These applications have undergone remarkable development in the past decade, driven by advances in machine-learning algorithms such as support-vector machines (Cortes and Vapnik 1995) and, most recently, convolutional neural networks (Lecun et al. 1998). Such progress is evident in recent object-recognition benchmarks (Russakovsky et al. 2015).

For most classification methods, accuracy depends on the degree of discrimination of the measured characteristics (i.e., features) with which the classifier is trained. Indeed, even the best classifiers will produce poor results when trained with features that do not represent well the target classes. Feature descriptiveness affects most classification methods that use pre-designed features.

Approaches for selecting features include feature selection (Pudil, Novovičová, and Kittler 1994) and dimensionality reduction (Jain, Duin, and Mao 2000). Feature selection finds the features that maximize an objective function of classification over the training data. In this approach, the selection of features is coupled with the classification technique. In contrast, dimensionality reduction extracts a large collection of features regardless their discriminative ability. These features are used by dimensionality reduction to find some underlying structuring of the data which is often of a lower dimensionality than that of the original feature set. With the learned structure at hand, the original high-dimension features are then mapped onto low-dimension features that preserve some of the original dataset’s structure in terms of class separation. Popular dimensionality-reduction techniques include principal component analysis (PCA) and linear discriminant analysis (LDA) (Duda, Hart, and Stork 2000).

Enhancing the discriminative power of features can also be done by incorporating high-level semantic information into the feature vectors. Here, feature vectors are augmented with extra components that describe contextual or semantic information of objects to be classified. Common semantic information added to features include spatial and temporal contexts. Ullah, Parizi, and Laptev classified human actions from videos by augmenting a bag-of-features classifier with context information in the form of segmented regions such as objects, roads, and sidewalks. Their feature augmentation improved classification by disambiguating local spatio-temporal features. Bettadapura et al. also augmented the bag-of-features framework with temporal sequencing information as well as spatial context. Their method classified activities such as road traffic, manual handing of surgical instrumentation, and soccer-player activities. Chen et al. incorporated classification rates of surrounding objects in an image to boost the classification of a target object, a process called iterative contextualization.

In this paper, we propose a method for increasing the discriminative power of features by augmenting them with high-level semantic information learned from training data (Section 2). Our method combines the advantages of dimensionality reduction techniques such as PCA and LDA with those of feature-selection and augmentation techniques. Instead of explicitly augmenting feature vectors, we transform them through a mapping function learned using a modified neural network that has been trained to categorize the data into target groups. Here, our idea is to use the memory of the neural network as a mapping function to embed categorization information into feature vectors. This mapping
function is obtained by removing the activation functions of the network’s last layer, leaving only its output in the form of weights. When low-level feature vectors are input to this modified network, the output of the modified last layer becomes the new mapped feature vectors that implicitly contain the high-level categorization information.

We tested our method on three classification tasks: pollen classification, action classification, and acoustic classification. In all tests, our feature-enhancing method increased classification accuracy (Section 3).

2 Proposed Method

Our goal is to incorporate application-specific semantic information into the feature vectors used for classification. Examples of types of semantic information include categorization, spatial and temporal context, and object use. Here, our intuition is that the weights of a neural network that is trained to categorize a set of semantic categories represents a mapping from a lower-level feature space to a higher-level semantic-weighted feature space. Once this mapping is at hand, general feature vectors extracted from the objects to be classified can be mapped through the neural network into semantic-weighted features (i.e., features that are implicitly weighted by the learned categorization information). In our method, the mapped features used for training and classification are the last layer of weights produced by our categorization neural network after its training is complete. Given the semantic-weighted features, classification is done using a support-vector machine classifier.

Our classification method’s main steps are as follows. First, a set of pre-categorized training data is selected for creating the neural-network category-relevant mapping. The set of training data is manually divided into groups representing semantic categories. The idea is for the network to learn an implicit representation of categories. This representation is then used to map lower-level features from the training data into a semantic-weighted feature space. Secondly, we use the features mapped through the neural network to train a classifier to perform recognition.

Next, we describe our method using a motion-classification task as the main example. We then show how the method can be used for classifying a dataset of pollen images, and a dataset of acoustic calls of frogs.

2.1 Feature Extraction

We begin by assuming the availability of a set of videos of basic human motions grouped into $k$ pre-defined motion categories, $V = \{v_j\}_{j=1}^n, i = 1, \ldots, k$. The selected categories are assumed to be distinct, and chosen to cover a number of real-world situations. For each video in $V$, we extract a set of lower-level spatio-temporal features using the method proposed by Dollar et al. (Dollár et al. 2005). Other spatio-temporal descriptors could also be used (Laptev 2005; Ke, Sukthankar, and Hebert 2005; Oikonomopoulos, Patras, and Pantic 2006). These features describe the information content inside small subvideos (e.g., cuboids of pixel intensity values) automatically extracted at spatio-temporal locations that are detected across the video volume. These locations correspond to spatio-temporal corners across the video volume (Laptev 2005). For example, the cuboid’s content can be described in terms of its optical flow, pixel intensity co-occurrence statistics, principal components, etc. Dollar et al. proposed a number of spatio-temporal features that can be used for motion recognition. In this paper, we use the brightness gradient and optical-flow features proposed by (Dollár et al. 2005). These features are usually less sensitive to noise caused by changes in color and illumination.

The set of extracted spatio-temporal features is generally quite sparse, and can occur at different locations in the video volume. Additionally, the total number of features can vary significantly from video to video, even for similar motions. In order to obtain a fixed-size descriptor for each video, we follow (Dollár et al. 2005) and build a frequency histogram of learned prototypical features for each video in $V$. This is a vector quantization method based on a vocabulary of prototypes learned using a k-means clustering algorithm. The prototype features are the centers of the K-Means clusters of features extracted from all videos in $V$. The K-Means clustering is performed for each class of features (i.e., brightness gradient and optical flow). Let $P = (p_1, \ldots, p_n, p_{n+1}, \ldots, p_N)$ represent a vector of prototype vectors (i.e., K-Means cluster centers). The first $n$ components of $P$ correspond to centroids from the brightness gradient clusters. The reminder components are centroids from the optical flow clusters. The total number of clusters for each feature type is provided by the user as an input to the algorithm. Based on the set of feature prototypes in $P$, we can label each extracted feature with the label of its closest prototype. This labeling can be accomplished by a simple nearest-neighbor classification procedure. Once the labeling data is at hand, we can represent the motion information in the video by a frequency histogram of its prototype labels. Let $f_k^i = (f_{j1}^i, f_{j2}^i, \ldots, f_{jn}^i)$ be the descriptor for a given motion sequence, where $f_{jn}^i$ is the...
Let \( f \) be a network. We associate each feature vector \( a \) to \( G \) in a way that its application to \( f \) produces a result similar to \( a^j \), where:

\[
a^j_k = \begin{cases} 
0, & k \neq j, \\
1, & k = j.
\end{cases}
\] (1)

Let \( s^{(q)}_j \) denote the output signal of the \( j \)-th neuron in the \( q \)-th layer, and \( w^{(q)}_{ij} \) the connection weight coming from the \( i \)-th neuron in the \( (q-1) \)-th layer to the \( j \)-th neuron in the \( q \)-th layer. More formally,

\[
s^{(q)}_j = \sigma(y^{(q)}_j) \quad \text{and} \quad y^{(q)}_j = \sum_{l=0}^{n_{q-1}} w^{(q)}_{ij} s^{(q-1)}_l,
\] (2)

where \( y^{(q)}_j \) is the activation level of the neuron, \( n_{q-1} \) is the number of neurons in the \( q-1 \)-th layer, and \( \sigma \) is the sigmoid activation function given by:

\[
\sigma(y) = \frac{1}{1 + e^{-y}}.
\] (3)

Given an input vector \((1, f^1, f^2, f^3, ..., f^N_{N-1})\) representing a video \( i \) in the neural network’s input layer (i.e., 0-th layer), the output signal of the \( j \)-th neuron in the \( L \)-th output layer is given by:

\[
s^{(L)}_j = \sigma \left( \sum_{m_L=0}^{N_{L-1}} w^{(L)}_{m_L,j} \sigma \left( \sum_{l=0}^{N_{l-1}} w^{(1)}_{l,m_l} s^{(1)}_l \right) \right)
\] (4)

A learning rule is applied using the network outputs, \( s^{(L)}_j \), for the new weights generation in \( G \). In our implementation, the backpropagation learning rule is used to adjust these weights until the network reaches a sufficient level of correctness in relation to the labels of the training database videos. Once this criteria is satisfied, \( G \) is considered to be trained, and its memory is loaded with pertinent semantic information provided by the pre-categorized training dataset.

We modify the structure of the trained network by removing the activation function \( \sigma(y) \) in the last layer. As a result, given low-level feature sets, \( f_i \), extracted from videos containing general human motions as input, this modified network will produce an output vector \( s_i = (s_1, s_2, s_3, ..., s_K) \) that will serve as the new high-level feature vector. This new feature vector carries implicit semantic information based on the pre-categorized groups used to train the neural network.

### 2.3 Classification

Our main goal is to show that our implicit semantic mapping procedure is able to carry important information that helps motion classification. To accomplish this goal, we test our feature mapping idea using a simple nearest-neighbor classifier (Duda, Hart, and Stork 2000).

### 3 Experiments

#### 3.1 Motion classification

**Feature Extraction.** We use the spatio-temporal features proposed by Dollar et al. (2005). These features are calculated based on the information contained inside small sub-videos (e.g., cuboids) extracted at spatio-temporal locations. Dollar et al. proposed spatio-temporal descriptors that are useful for motion recognition. We use videos with frame size of 180 × 144 pixels captured at a rate of 50 fps. We use cuboid size of 20 × 20 pixels and 7 frames depth.

Pre-categorizing these videos is a hard problem. In this paper, we limited our categorization set to contain videos of human motion only. These videos were obtained from the Weizmann human action dataset (Blank et al. 2005a).

We used two spatio-temporal features: intensity gradient and optical flow. We extract 25 cuboids for each video. From each cuboid, we calculated the intensity gradient and optical flow. These features are distributed at different locations in each video. We used vector quantization to create histograms representing each video. The prototype features (i.e., histogram bins) are chosen to be the means of clusters obtained using the k-means algorithm.

**Datasets.** Here, we evaluated our method on the Weizmann human action dataset (Blank et al. 2005a). The Weizmann dataset contains nine action classes performed by nine different subjects. Figure 2 shows sample video frames from the dataset.
Data Preparation. For the experiment, we collected a database of 90 low-resolution (180 × 144, deinterlaced 50 fps) video sequences with human activities. The human activity data comes from the dataset collected by (Blank et al. 2005b) which has become a standard test database for similar action recognition tasks. There are 9 individuals each performing 10 natural actions such as run, walk, jumping-jack (or shortly jack), jump-forward-on-two-legs (or jump), jump-in-place-on-two-legs (or pjump), gallop-sideways (or side), wave-two-hands (or wave2), wave-one-hand (or wave1), or bend. This dataset contains videos with static camera and simple background. Some example frames are shown in Figure 2.

We used the spatio-temporal cuboids as network inputs and after several tests performed, we chose the gradient and optical flow descriptors. The number of clusters used to form the cuboids vocabulary was chosen in an empirical form and resulted in k = 800 for the Optical Flow feature and k = 800 to the feature gradient. The initial positions of the cluster centers are chosen randomly.

The neural network implemented in this work has 1600 neurons in the input layer (bias and low-level features), 80 neurons in the hidden layer and 10 neurons on the output layer (high-level features), i.e., the network has a 1600-80-10 structure. For the training, the learning rate was set at 0.0003 and the momentum was set at 0.75.

To test and train the network, we used the leave-one-out strategy. This was done by selecting a set of videos from the dataset and leaving them out of the training system. We removed the entire sequence from the database while other actions of the same person remain. To calculate an average, we iteratively leave out each set of video for all individuals from the database. In this manner, we are able to protect against biases that may arise from using any single video.

The results are summarized in Figure 3. Overall, the results were promising. The majority of the action classes were correctly classified. In particular, the Bend, Jack, Pjump, Walk actions were classified with 100% accuracy. The proposed approach performed well both when the individual remained in the same position or when she moved across the scene.

Also, the method performed well on most of the actions except ‘jump’, ‘run’ and ‘skip’. These two actions are very similar to each other in the way that the actors bounce across the video. The Skip movement was the most difficult action to classify, so some works did not include in the results table. The method misclassified those actions where the pose does not change significantly during the motion (e.g., wave1 and wave2 actions). Considering that the semantic concept searches are highly complex, the obtained results were effective despite the fact that it is difficult to differentiate movements through some changes in behavior. Our best obtained precision was 85.7% and reduces in 99.37% the vector dimensionality, such that, for large video databases, this fact could mean a big reduction in the processing time.

We compare our results to Goodhart et al. (2007), Scovanner et al. (2007) and Niebles and Fei-Fei et al. (2007). A comparison between ours and other approaches is presented in the Table 1. The results demonstrate that our performance is better than other known methods.

Table 1: Comparison of different methods using Weizmann Human Action dataset.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our Method</td>
<td>85.7</td>
</tr>
<tr>
<td>Goodhart et al.</td>
<td>84.6</td>
</tr>
<tr>
<td>Scovanner et al.</td>
<td>82.6</td>
</tr>
<tr>
<td>Niebles and Fei-Fei et al.</td>
<td>72.8</td>
</tr>
</tbody>
</table>

3.2 Pollen classification

We performed the bag-of-word technique to extract features of 30 types of pollen grain images. Surf features are extracted in a grid across the input image, we have 1063 optical image. Then, we create our codeword by performing clustering process using k-means with 500 clusters. After that we code pollen grain images using the codeword to create histogram as determinative features. Neural network is trained based on the histogram of the input images. The recognition rate was ≈80% based on the direct classification using neural network. In the next step of this work is to use the trained neural network to convert the histogram features from space to another. The neural network uses the learned weights to project the histogram features to high level features. Finally, we train SVM based on the projected features to perform the classification. The results showed that we improved the classification rate to gain ≈85% after we mapped the features using neural network. This mapping process is very similar to the all techniques that reduce the dimensionality by transforming the features from low level to high level representation.
We presented an approach for enhancing the discriminative
with mapping: 73.58%

We extract MFCC at these high peaks fear each calls. Then
recognition. After we transom the frog call samples to spec-

Based on spectrogram, we tested the proposed mapping
techique using 216 frog calls of 15 species to perform frog

Classification rates: Classification rate using standard fea-
tures was 79.62%. Classification rate using mapped features
was 84.91

Table 2: Evaluation Measurements of pollen recognition

<table>
<thead>
<tr>
<th>Method</th>
<th>Precision</th>
<th>Recall</th>
<th>sensitivity</th>
<th>specificity</th>
<th>F score</th>
</tr>
</thead>
<tbody>
<tr>
<td>The direct method</td>
<td>81.07 %</td>
<td>78.85 %</td>
<td>78.85 %</td>
<td>99.30 %</td>
<td>79.13 %</td>
</tr>
<tr>
<td>Mapping method</td>
<td>87.41 %</td>
<td>83.89 %</td>
<td>83.89 %</td>
<td>99.48 %</td>
<td>84.47 %</td>
</tr>
</tbody>
</table>

3.3 Acoustic classification

Based on spectrogram, we tested the proposed mapping
 technique using 216 frog calls of 15 species to perform frog
 recognition. After we transom the frog call samples to spec-
trogram, we detect the high peaks of the spectrogram. Then,
we extract MFCC at these high peaks fear each calls. Then
we perform bag of features to quanitze these MFFC fea-
tures. At the start, we take the MFCC columns correspond-
ing to the location of the high peaks. Then we cluster the
columns to find prototypes of high-density regions. After
that we select the centroid as prototypes representing each
cluster. Finally, we use vector quantization to relabel the
MFCC column vectors to form fixed-length feature vectors
as histogram features.

Then we use neural network to map the histogram feature.
note that we divide our data set into 75

Classification rates. Direct features: 66.04%. Features
with mapping: 73.58%

Table 3: Evaluation Measurements of frog recognition

<table>
<thead>
<tr>
<th>Method</th>
<th>Precision</th>
<th>Recall</th>
<th>sensitivity</th>
<th>specificity</th>
<th>F score</th>
</tr>
</thead>
<tbody>
<tr>
<td>The direct method</td>
<td>60.89 %</td>
<td>59.67 %</td>
<td>59.67 %</td>
<td>97.57 %</td>
<td>58.67 %</td>
</tr>
<tr>
<td>Mapping method</td>
<td>79.89 %</td>
<td>73.22 %</td>
<td>73.22 %</td>
<td>98.11 %</td>
<td>72.58 %</td>
</tr>
</tbody>
</table>

4 Conclusions

We presented an approach for enhancing the discriminative
power of features in classification tasks. Our method uses

the last layer of a pre-trained neural network as a mapping
function to transform raw features into category-enhanced
features. This network embeds categorization information.
The method was tested on three main classification tasks,
namely, human-action classification, pollen-grain classifica-
tion, and acoustic classification. Our results were promising
and show that the mapping improves classification.

References

Bettadapura, V.; Schindler, G.; Ploetz, T.; and Essa, I. 2013. Augmenting bag-of-words: Data-driven discovery of temporal
IEEE Computer Society.

1402.

Computer Vision (ICCV’05), 1395–1402.


In VS-PETS.


Ke, Y.; Sukthankar, R.; and Hebert, M. 2005. Efficient visual event detection using volumetric features. In ICCV’05:
Proceedings of the Tenth IEEE International Conference on Computer Vision (ICCV’05) Volume 1, 166–173.


