HAL
open science

A Deep Learning Framework for Tactile Recognition of Known as well as Novel Objects

Zineb Abderrahmane, Gowrishankar Ganesh, André Crosnier, Andrea Cherubini

To cite this version:

Zineb Abderrahmane, Gowrishankar Ganesh, André Crosnier, Andrea Cherubini. A Deep Learning Framework for Tactile Recognition of Known as well as Novel Objects. IEEE Transactions on Industrial Informatics, 2020, 16 (1), pp.423-432. 10.1109/TII.2019.2898264 . hal-02012628

HAL Id: hal-02012628

https://hal.science/hal-02012628v1

Submitted on 8 Feb 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

A Deep Learning Framework for Tactile Recognition of Known as well as Novel Objects

Zineb Abderrahmane, Gowrishankar Ganesh, André Crosnier, and Andrea Cherubini

***Abstract*—This paper addresses the recognition of daily-life objects by a robot equipped with tactile sensors. The main contribution is a deep learning framework that can recognize objects already touched as well as objects never touched before.**

To this end, we train a Deconvolutional Neural Network that generates synthetic tactile data for novel classes. Then, we use both these synthetic data and the real data collected by touching objects, to train a Convolutional Neural Network to recognize both known (trained) objects and novel objects. Furthermore, we propose a method for integrating newly encountered data into novel classes. Finally, we evaluate the framework using the largest available dataset of tactile objects descriptions.

***Index Terms*—Tactile object recognition, Deep learning, Zero-Shot Learning, One-Shot Learning, Convolutional Neural Networks, Generative Adversarial Networks.**

I. Introduction

ROBOTS operating in household environments are surrounded by a wide variety of objects. Just like a blind person, a robot can use its tactile sensors to recognize objects when vision is not available (e.g. when performing in the dark). In this work, we study the recognition of daily-life objects using the sense of touch.

Traditionally, in the tactile recognition literature [1]–[8], robots are trained with the data recorded while touching multiple objects. Such data are then mapped to a finite set of training object classes. Finally, this mapping is used to recognize unknown objects. This method however, omits an important case: the recognition of novel objects not included in the training set, which is often the case in daily life. This is because, first, the number of objects in common daily environment can be huge (30000 as estimated in [9]), and second, new objects being added continuously. It is therefore impossible to train a robot on all daily-life objects in advance.

The first solution that comes to mind is to retrain the robot each time it encounters a novel object. However, tactile data collection is non-trivial, because the robot control required to guarantee stable and complete robot-object physical interactions can be very complex, and the use of tactile sensors involves to satisfy temporal constraints in order to guarantee the quality of acquired data. Due to these reasons, a solution that aims to collect training tactile data each time a novel object is encountered can hamper robot operation.

A solution to the above problems, which drastically decreases the cost of novel objects recognition, is to collect *semantic* information about objects, or specifically, human-understandable properties [10]. These can be provided by humans [11] or automatically mined from semantic databases (e.g. Wikipedia [12]). The semantic information can then be used with the tactile data available from trained objects to recognize new ones.

A popular form of semantic information is attributes, which refer to human-comprehensible semantic object properties. Attributes like *thin*, *long*, *wooden*, *tapered*, *metallic* and *rubber*, can describe a pencil. If a robot has learned the relation between attributes and tactile data, then given the attribute-based description of a pencil, the robot can recognize a pencil when touching it even if it has not touched any pencil during the training phase. This approach is called *attribute-based Zero-Shot Learning* (ZSL) [11].

In this work, we design a unified recognition framework that, from tactile data, can recognize an object as a known (previously touched) or as novel object. Recognition of known objects is performed by using available multi-class classifiers built from training data, while recognition of novel objects relies on attribute-base ZSL approach. Furthermore, the integration of new tactile data for novel objects, starting from only one training sample, can be achieved by performing One-Shot Learning (OSL).

The proposed framework is based on deep Convolutional Neural Networks (CNNs). This choice was motivated by the good performance reported recently in multi-class tactile recognition [7], [8] when using CNNs. However, note that the use of CNNs for Zero-Shot Learning is not straightforward: if we simply train a CNN to map tactile data into object classes, the CNN will miss output classes having no training data. This decreases the classification accuracy due to the imbalance in the training set [13], [14]. To cope with this problem, for each novel class specified by its attribute-based description, we generate synthetic tactile training data.

This paper is structured as follows: in the next section, we present related studies on tactile recognition and ZSL, in addition to our contributions with respect to these studies. We give an overview of the proposed recognition framework in Sect. III. The details about the training data generation, and about the ZSL and OSL are presented in Sect. IV. The experimental setup is presented in Sect. V. Finally, we present and discuss the evaluation results in Sect. VI, and we give the conclusions in Sect. VII.

II. Related Work

A. Multi-Class Tactile Recognition

Following the massive advance in visual recognition [15], many works have studied the integration of new data sources such as tactile sensing, to recognition systems. This allowed the integration of new physical properties that cannot be estimated using vision, such as material [4], compliance [16]

All authors are with the Interactive Digital Human IDH group, Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier (LIRMM), University of Montpellier CNRS, 860 Rue Saint Priest, 34095 Montpellier, France. E-mail:firstname.lastname@lirmm.fr.

G. Ganesh is also with the CNRS-AIST JRL UMI3218/RL 1-1-1 Umezono, 305-8560 Tsukuba, Japan. E-mail: gans_gs@hotmail.com

and texture [17] for object recognition. Several works have been proposed to deal with challenges associated with tactile data processing, including data sparsity and noise [18], choice of object sets, and the sensing capabilities and constraints of robots [2] and using techniques such as deep learning [1] and sparse coding [19]. Recent advances in tactile object recognition are reviewed in [20]. However, all these methods train and test their recognition systems on the same set of object classes. Next, we review studies focusing on Zero-Shot Learning, for recognizing novel objects having no training data.

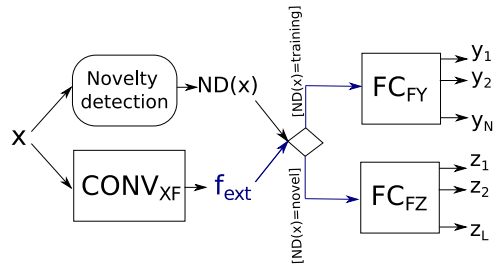


Fig. 1. Overview of our framework: recognition of known and novel objects.

B. Visual Zero-Shot Learning

Recognition of novel object classes without training examples, known as Zero-Shot Learning (ZSL), has gained increasing interest in the visual recognition community. Recognition of novel objects is possible when semantic descriptions of object classes are available. This allows to mine relationships between training and novel objects, and hence to use training data to recognize novel classes. A popular approach is to determine object properties, or *attributes* [10], [21]–[27]. Textual descriptions extracted from Wikipedia articles [12], [28] form a second popular source of semantic information. A third approach directly mines objects relationships from Linguistic knowledge databases such as WordNet [29], Wikipedia and search engines [30].

The semantic descriptions have been used in different ways to achieve ZSL. Authors of [31] solved ZSL by predicting regression parameters of a one-vs-all classifier for each novel class. Direct and hierarchical similarity measures have been used in [32] to compute the classification score of each novel class depending on scores of *similar* known classes. The works that are closest to ours aim at generating data samples for novel classes from their semantic descriptions. Authors of [33] learned a kernel-based regressor that predicts a prototypical visual sample for each novel class, based on its representation in a semantic space. In [34], a variational auto-encoder was used to generate data for novel classes based on their attributes. Authors of [35] proposed an embedding algorithm to generate synthesized data from semantic features. A series of approaches to perform ZSL is reviewed and compared in [36].

C. Extension to One-Shot Learning

An extension of ZSL, known as One-Shot Learning (OSL), is when only one tactile sample become available for novel classes. The recognition system usually under-fits these classes because of the very few number of samples compared to other classes. Few works extend ZSL to OSL. An example is [21], which applies topic modelling to attributes in order to recognize objects having zero or one training sample each. Another example is [37], in which prototypical networks are used for both OSL and ZSL.

D. The contribution of this paper

The difficulty of tactile data collection makes classes with zero or one training sample very frequent in training sets. However, ZSL and OSL have gained significantly less attention in tactile recognition, compared to vision. Haptic ZSL was studied in [38] using Direct Attribute Prediction [11] to recognize 10 novel daily-life objects with a robotic hand with tactile sensors. Authors of [39] performed material-based tactile ZSL, using visual training data available for novel classes. Texture-based OSL was performed in [40] using least squares support vector machines to recognize 12 new textures given only one tactile training sample per texture.

Here our main contributions are as follows:

1) We propose a comprehensive tactile recognition system, that can recognize classes having many training data as well as classes without or with very few training data, which is very common in daily-life tactile recognition.
2) Recognition of novel objects using their attribute-based description and tactile data collected from training objects. This is a very important advantage in tactile recognition as it reduces tactile data collection and replaces it with semantic information which is much easier to obtain.

III. Tactile Object Recognition Framework

A. Problem Statement

We study the problem of training a robot on recognizing the set of object classes $Y = \{y_1, \ldots, y_N\}$ from tactile data and testing it on recognizing the set $O = Y \cup Z$ including both training classes Y and novel classes $Z = \{z_1, \ldots, z_L\}$, s.t. $Y \cap Z = \emptyset$. During the training phase, the robot collects tactile data from each object $y \in Y$ ensuing the training set $D_{train} = \{(y_n, (x_1, \ldots, x_{In})), n = 1, \ldots, N\}$, where I_n is the number of training samples collected from y_n, and all tactile samples x_i are represented in a certain space X. During the test phase, a tactile sample $x \in X$ should be classified as one of the set O objects without any prior knowledge if it belongs to a training object (to Y) or to a completely novel one (to Z) and without collecting any additional training data from any of Z objects.

B. Recognition Framework Overview

Fig. 1 illustrates our proposed tactile recognition framework capable of recognizing both training and novel objects. To recognize a tactile test sample $x \in X$, this is processed by a convolutional network $CONV_{XF}$ that extracts a features vector $f_{ext} \in F$, where F is the space in which the CNN features are represented. This latter is classified by one of two fully connected neural networks, FC_{FY} or FC_{FZ}, according to a novelty detection metric $ND(x)$. This metric predicts if x is novel or not; if x is novel, then f_{ext} is classified using FC_{FZ}, having only objects in Z as outputs. If x has been collected from a training object, then f_{ext} is classified using FC_{FY},

classifying objects in Y only. This architecture requires to define a novelty detection metric and to use the set D_{train} to train convolutional and fully connected networks.

On one hand, we propose to use a Gaussian Mixture Model (GMM) as a novelty detection metric. We use GMM to estimate the density distribution of the tactile training data from D_{train}. A data sample x is classified as *novel* if it belongs to a region in the input space with low density, and as *training* otherwise. Formally, we compute a weighted log-likelihood of the fitted GMM given x: if it is lower than a threshold σ_{nov}, then x is *novel*.

On the other hand, D_{train} is the only training set we have to train the convolutional and fully connected networks in the framework. If we consider the CNN consisting of $CONV_{XF}$ and FC_{FY}, it can directly be trained using D_{train} since it maps X into Y. However, a problem arises with training FC_{FZ} since D_{train} does not include any tactile data collected from Z. Next, we present how we proceed to train FC_{FZ} without collecting additional tactile data, other than D_{train}.

C. Training FC_{FZ}

Our solution to train FC_{FZ} without collecting any tactile data other than D_{train} is to generate synthetic training data for each $z \in Z$. This requires acquiring semantic information about objects. By learning the relationship between semantic and tactile spaces, synthetic tactile data can be generated for each object based on its semantic description.

Following the success of attribute-based ZSL, we choose attributes as a popular, efficient and intuitive semantic representation of objects. Let us consider the set of attributes $A = \{a_1, \ldots, a_M\}$. We describe each object $o \in O$ with a deterministic attribute vector $\mathbf{a^o} = (a^o_1, \ldots, a^o_M)$, where for each $m = 1, \ldots, M$: $a^o_m = 1$, if a_m is present for object o and $a^o_m = 0$ otherwise. Let us take the example of $O = \{$pencil, bottle, mug$\}$, and $A = \{$wooden, glass, porcelain, cylindrical, thin, concave$\}$. We can describe objects in O using the following attribute vectors: $\mathbf{a^{pencil}} = [1, 0, 0, 1, 1, 0]$, $\mathbf{a^{bottle}} = [0, 1, 0, 1, 0, 0]$ and $\mathbf{a^{mug}} = [0, 0, 1, 1, 0, 1]$.

Then, our solution for generating synthetic data to train FC_{FZ} is to learn a generator $G : A \longrightarrow F$ that generates a feature vector in F given an attribute vector in A. Once G is learned, G generates for each $z \in Z$ a set of feature vectors $f_{gen} = G(\mathbf{a^z})$ using $\mathbf{a^z}$. Finally, FC_{FZ} is trained on classifying f_{gen} as z.

IV. Generating Synthetic Features for Novel Objects

A. Solution Overview

We summarize in the following steps our solution for training the generator G:

1) We train the CNN consisting of $CONV_{XF}$ and FC_{FY} on classifying objects in Y using D_{train};
2) We train a Deconvolutional Neural Network G on generating synthetic features $f_{gen} \in F$ using attribute vectors $\{\mathbf{a^{y_1}}, \ldots, \mathbf{a^{y_N}}\}$ describing training objects;
3) We improve the quality of G to generate features f_{gen} similar to those extracted from real tactile data f_{ext}.
4) We use the trained G to generate for each $z \in Z$ a set of I_z synthetic features $F_z = \{f_{gen,1}, \ldots f_{gen,I_z}\}$ using $\mathbf{a^z}$.

Next, we detail our proposed solution to perform each step.

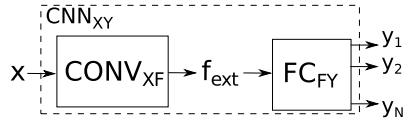


Fig. 2. Classification of training objects using CNN_{XY}: $CONV_{XF}$ is the convolutional part and FC_{FY} is the fully connected part.

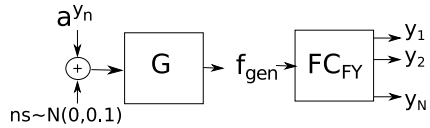


Fig. 3. Train G to generate features associated with objects in Y.

B. Classifying Training Objects

The first step is to use D_{train} to train $CONV_{XF}$ and FC_{FY} to map tactile samples from X to Y classes. We remove from our framework illustrated in Fig. 1, the novelty detection and FC_{FZ} since we are working with Y only. This is equivalent to training the CNN illustrated in Fig. 2 using D_{train}.

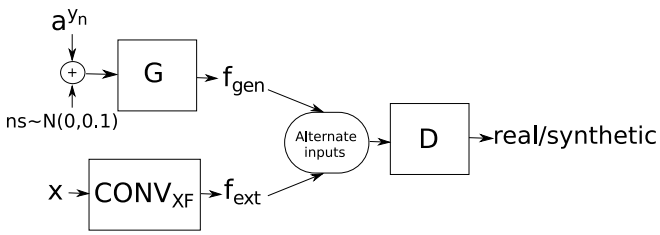
C. Training a Synthetic Features Generator

The second step is to train a Deconvolutional Neural Network G to generate synthetic features in F from attributes in A. As illustrated in Fig. 3, we use the pre-trained FC_{FY} to train G on how to generate features $f_{gen} \in F$ corresponding to attributes $\mathbf{a^{y_n}}$ from $D^a_{train} = \{(y_n, \mathbf{a^{y_n}}), n = 1, \ldots, N\}$. Here, FC_{FY} is not fine-tuned and thus its parameters are not updated. Training G is described in **Algorithm 1**: for each pair $(y_n, \mathbf{a^{y_n}})$ (line 3), $\mathbf{a^{y_n}}$ is input to G after adding random noise, to generate f_{gen} (lines 4, 5). The random noise serves to generate multiple feature vectors for the same attribute vector at different training epochs. The generated f_{gen} is input to FC_{FY} that classifies it as $y_{pred} \in Y$ (line 6). Actually, the goal of G is to generate from $\mathbf{a^{y_n}}$ features f_{gen} that are classified by FC_{FY} as y_n. This comes down to minimizing the loss L_{FC} between predicted y_{pred} and desired y_n, computed at line 7. The parameters θ_G of G are updated using the Adam optimization algorithm [41], where $\partial L_{FC}/\partial\theta_G$ is the gradient of L_{FC} with respect to θ_G (line 8).

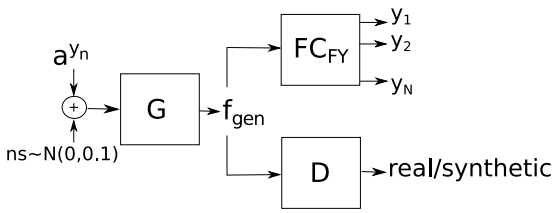
Algorithm 1: Training G

Input: D^a_{train}, number of training epochs $epoch_{max}$, number of training objects N, pre-trained FC_{FY}
Output: Trained G

1: **for** $epoch_i = 1$ to $epoch_{max}$ **do**
2: **for** $n = 1$ to N **do**
3: $(\mathbf{a^{y_n}}, y_n) \leftarrow Next(D^a_{train})$ {Load the next pair}
4: $ns \sim \mathcal{N}(0, 0.1)$
5: $f_{gen} \leftarrow G(\mathbf{a^{y_n}} + ns)$
6: $y_{pred} \leftarrow FC_{FY}(f_{gen})$
7: $L_{FC} \leftarrow loss(y_n, y_{pred}))$
8: $\theta_G \leftarrow Update(\theta_G, \partial L_{FC}/\partial\theta_G)$
9: **end for**
10: **end for**
11: **return** G



(a) Train D to distinguish between real and generated features.



(b) Train G to generate synthetic features similar to real ones.

Fig. 4. Adversarial training of G and D.

D. Generating Realistic Features

Third, since our goal is to train FC_{FZ} using generated features and test it using real ones, we must improve the quality of the generated features to make them "as similar as possible" to the real ones. To this end, we continue training G by adding another convolutional network, called D, that discriminates between synthetic and real features. G and D are trained via an adversarial process illustrated in Fig. 4 and detailed in **Algorithm 2**: at each training iteration, we input noised $\mathbf{a^{y_n}}$ to G to obtain feature vector f_{gen} (lines 5-7) and a real tactile sample x to $CONV_{XF}$ to extract f_{ext} (lines 8, 9). Then, we train D and G alternately, such that we train D (resp. G) in Fig. 4a (resp. Fig. 4b) using G (resp. D) parameters obtained from the previous step and without fine-tuning G (resp. D). We start with training D (see Fig. 4a) on returning *'synthetic'* when inputting f_{gen} and *'real'* when inputting f_{ext} (lines 11-16). We keep fine-tuning D until its loss becomes lower than a certain threshold σ_D (lines 17-19). Then, we switch to fine-tuning G using the newly trained D and the pre-trained FC_{FY} simultaneously (see Fig. 4b). We update the G parameters, on one hand to minimize the loss between the desired y_n and predicted y_{pred} by inputting f_{gen} into FC_{FY} (lines 22-24). On the other hand, the updated G should generate f_{gen} that D erroneously classifies as *'real'* (lines 25-27). We keep training G until the losses of FC_{FY} and D become lower than a certain threshold σ_G (lines 28-30). Then, we go back to training D with the new updated G. We continue alternating between training D and G (Figures 4a and 4b, respectively). This adversarial training converges when D becomes unable to distinguish anymore between real and generated synthetic features. This means that G is generating synthetic features that are indistinguishable from real ones.

Algorithm 2: Adversarial training of G

Input: G trained using algorithm 1, D^a_{train}, D_{train}, pre-trained FC_{FY}, number of training epochs $epoch_{max}$, number of training objects N,σ_D: threshold of D training loss, σ_G: threshold of G training loss

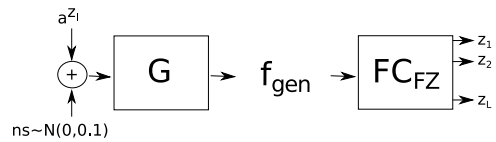


Fig. 5. Train FC_{FZ} using data generated by G.

Output: Trained G to output realistic features

1: $train_G \leftarrow False$
2: $train_D \leftarrow True$ {Start with training D}
3: **for** $epoch_i = 1$ to $epoch_{max}$ **do**
4: **for** $n = 1$ to N **do**
5: $ns \sim \mathcal{N}(0, 0.1)$
6: $(\mathbf{a^{y_n}}, y_n) \leftarrow Next(D^a_{train})$
7: $f_{gen} \leftarrow G(\mathbf{a^{y_n}} + ns)$
8: $x_i \leftarrow Next(D_{train})$
9: $f_{ext} \leftarrow CONV_{XF}(x_i)$
10: **if** $train_D$ **then**
11: $d_{pred} \leftarrow D(f_{gen})$ {See Fig. 4a}
12: $L_s \leftarrow loss(d_{pred}, 'synthetic')$
13: $d_{pred} \leftarrow D(real_feat)$
14: $L_r \leftarrow loss(d_{pred}, 'real')$
15: $L_D \leftarrow L_s + L_r$
16: $\theta_D \leftarrow Update(\theta_D, \partial L_D / \partial \theta_D)$
17: **if** $L_D/2 < \sigma_D$ **then**
18: $train_G \leftarrow True$
19: $train_D \leftarrow False$
20: **end if**
21: **else if** $train_G$ **then**
22: $y_{pred} \leftarrow FC_{FY}(f_{gen})$ {See Fig. 4b}
23: $L_{FC} \leftarrow loss(y_n, y_{pred})$
24: $\theta_G \leftarrow Update(\theta_G, \partial L_{FC} / \partial \theta_G)$
25: $d_{pred} \leftarrow D(f_{gen})$
26: $L_D \leftarrow loss(d_{pred}, 'real')$
27: $\theta_G \leftarrow Update(\theta_G, \partial L_D / \partial \theta_G)$
28: **if** $L_{FC} < \sigma_G$ and $L_D < \sigma_G$ **then**
29: $train_G \leftarrow False$
30: $train_D \leftarrow True$
31: **end if**
32: **end if**
33: **end for**
34: **end for**
35: **return** G

E. Generating Training Data for Novel Classes

As illustrated in Fig. 5, once G is trained, we use it to generate a set of I_z synthetic features for each $z \in Z$. We input its associated $\mathbf{a^z}$ to the trained G, I_z times with different noise values. This generates a set $F_z = \{f_1, \ldots, f_{I_z}\}$ that will be considered as the synthetic training set of z.

The last step is to use $D^Z_{train} = \{(z_l, F_{z_l}), l = 1, \ldots, L\}$ to train FC_{FZ}. We refer to this method as GEN-F. A variant of this method replaces the synthetic features by real ones collected from Y. Given F_{z_l} for each $z_l \in Z$, we generate another training set for z_l by replacing each f_i by its nearest neighbor (using L1 distance) in F space from features extracted from real training data. Thus, each z_l is trained using data collected from objects in Y that are the most similar to F_{z_l} in F. We refer to this variant as NEIGHB-F.

F. Extension to One-Shot Learning

Our framework, trained on real data for Y and on generated data only for Z, can integrate new real data for objects in Z, which can become available with time. Here, we focus on the extreme case of OSL where one training sample arrives for each $z_l \in Z$. We obtain $D^Z_{train} = \{(z_l, x_l), l = 1, \ldots, L\}$. Directly integrating the only data sample available for each class is not expected to significantly improve the recognition performance, due to the tiny number of new samples. Instead, we use $CONV_{XF}$ to extract features of each x_l, yielding $f_l = CONV_{XF}(x_l)$. Then, we use the k nearest neighbors of each f_l in F to resume training FC_{FZ} for class z_l. In this case, each new sample for an object $z_l \in Z$ can improve training with k samples instead of only one sample.

V. Experimental Setup

A. Dataset

We evaluate our framework on the public PHAC-2 dataset [42] used by many state of art studies for tactile understanding [43]–[45]. This dataset contains 60 objects having a wide variety of texture, material and stiffness properties. Objects are described using a list of 24 binary *haptic adjectives*. After removing adjectives that are present in less than three objects, we obtained 19 adjectives that we use as attributes in our work: $A = \{absorbent, bumpy, compressible, cool, fuzzy, hard, hairy, metallic, porous, rough, scratchy, slippery, smooth, soft, solid, springy, squishy, textured, thick\}$.

These attributes were defined by human participants, who blindly explored objects using their hands, and expressed their sensations using words. Defining good quality attributes is still an open research problem. Two main points must be considered: (1) Defining non-ambiguous attributes that describe well the objects and increase objects separability and (2) reducing the effort of attributes definition and class-attribute labeling. For instance, thanks to crowd-sourcing, humans can collaborate to describe voluminous object datasets using haptic attributes. Clearly, this takes much less effort than exploring all objects using the robot. Studies such as [10] focused on this problem. However, this is out of the scope of this work, we take attributes provided with PHAC-2 dataset.

Authors of [42] explored each one of the 60 objects 10 times (trials) using the gripper of the Willow Garage PR2 Robot equipped with 2 SynTouch BioTac® sensors. In our work, we use data collected from the pair of BioTacs and we do not consider the gripper kinesthetic data. This is because of the simple and similar shapes of PHAC-2 objects. BioTacs readings are pre-processed and data are augmented following the method of [43] that used BioTacs readings for binary classification of all attributes in A. This consists in first transforming BioTac signals measured from each exploration trial into a tactile image of 32 channels $\times$ 30 time samples. The 32 channels correspond to the 4 pressure and temperature BioTac readings along with the first 4 principal components (obtained by PCA) of the 19 BioTacs electrode signals, all measured during 4 exploration procedures leading to $((4+4) \times 4 = 32$ channels). This defines the tactile data space $X = \mathbb{R}^{32\times 30}$. By considering the 2 BioTacs as identical and after augmenting data by sub-sampling the signals using 5 different starting points, each exploration trial ensues 10 samples (2 BioTacs $\times$ 5 signals). Thus, we obtain a raw tactile dataset composed of 6000 samples (60 objects $\times$ 10 trials $\times$ 10 samples).

TABLE I
Neural Networks' hyper-parameters used in our work.

Neural Network	Type	layers	Convolutional layers parameters			
			channels	stride	kernel	group
$CONV_{XF}$	conv.	2	96-256	2-2	(3,1)-(3,1)	32
G	deconv.	2	96-256	2-1	(4,1)-(3,1)	No
D	conv.	2	96-1	2-2	(3,1)-(3,1)	No

ZSL requires to split the 60 objects into 2 disjoint sets Z and Y. We randomly select 6 objects (10%) to be the test objects for Z and the remaining 54 objects for Y (90%). In order to ensure the framework's robustness to the choice of Y and Z, we repeat this splitting process 7 times to generate different splits Z-Y. This avoids reporting results that are dependent on the choice of objects rather than on the design of the solution. Finally, we made sure that spaces F and A were correlated for each split.

B. Implementation Choices

In Table I, we present the architecture of networks used in this work. Hyper-parameters were tuned to find a compromise between complexity and number of samples available to train each network. In our case, we have 100 samples for each object in Y, a number that does not allow us to train very complex models. FC_{FY} and FC_{FZ} are both one-layer fully connected networks. Convolutional Layers are followed by ReLU activation function for non-linearity. The weights of both the convolutional and fully connected layers are initialized using the Xavier method [46] and all deconvolutional layers are initialized using a Gaussian initializer. We used softmax function followed by multinomial logistic loss to train the fully connected layers and cross-entropy loss to train D. According to the architecture of $CONV_{XF}$, we obtained the features space $F = \mathbb{R}^{256\times 6}$, where 256 is the number of channels and (6×1) is the size of the output. Algorithms 1 and 2 are implemented by using batches of 50 samples.

All our algorithms are implemented in Python and executed on a PC with an Intel(R) Core(TM) i7-3840QM 2.8 GHz processor and a 8 GB RAM. We exploited the available code[1] developed in [43] to process and extract features from the PHAC-2 database raw data. Python scikit-learn[2] was used to estimate the parameters of the GMM. We used Caffe [47] to implement all networks listed in Table I. Finally, we used[3] [48] to implement Generative adversarial Networks (GAN) for the adversarial training of G and D. Finally, each convolutional network is trained for 600 epochs, which takes from 25 to 30 minutes, using the CPU only. Since we perform an offline recognition, we are not constrained by the training time, however, it can be optimized by using GPU.

VI. Evaluation

A. Object Splits

First, we analyze the characteristic of objects used in our experiments. Fig. 6 illustrates some examples of PHAC-2

[1] people.eecs.berkeley.edu
[2] sikit-learn.org
[3] github.com/samson-wang/dcgan.caffe

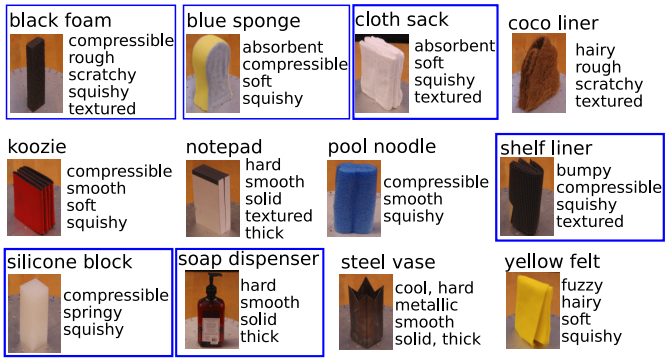


Fig. 6. Test objects (framed in blue) with their attributes (right side) in Z for split 1 and examples of training objects with their attributes (right side).

objects, their attributes, and the test objects of split 1. We note that, although test objects (framed in blue) are semantically different from training ones, both sets share the same attributes. For instance, the soap dispenser shares the attribute *smooth* with the koozie, all its attributes with the notepad and attribute *smooth* with the pool noodle. On the other hand, although test objects share some attributes, each one of them has a discriminative attribute vector that distinguishes it from the other. For instance, the silicone block and the blue sponge are both *compressible* and *squishy*, yet, the first is *springy* while the second is *absorbent* and *soft*. The shared attributes between the sets Z and Y and the uniqueness of the attribute vector of each object in Z are verified for each split, which allows to perform ZSL using our framework.

B. Novelty Detection

Given a test sample $x \in X$, we want to estimate whether it is collected from a training or a novel object using a GMM. To tune σ_{nov}, we split Y into two disjoint sets Y_{tr} and Y_{val}: Y_{tr} contains 90% of the training objects (48 objects), while Y_{val} contains the remaining 6 objects (10%). Then, we split the data collected from Y_{tr} into X_{tr} and X_{te}: X_{tr} contains 90 samples (90%) per object and X_{te} contains the remaining 10 samples (10%) per class. First, X_{tr} is used to fit the GMM, then we tune σ_{nov} to maximize the accuracy of classifying X_{te} as collected from training objects and X_{val} (collected from Y_{val}) as novel. Fig. 7 shows that very low threshold values classify the majority of samples as known and very high values classify all the samples as novel. Thus, we choose for each split the σ_{nov} that maximizes the average accuracy of classifying X_{te} samples as known and X_{val} samples as novel. Once σ_{nov} is tuned for each split, we report in Table II the accuracies in classifying X_{te} as known, and both X_{val} and X_Z (collected from Z) as novel. By averaging accuracies over all splits, we found that 90.3% of x collected from training objects and not used to fit the GMM, have been classified as known, and that 89.5% of data collected from novel objects have been classified as novel.

C. Multi-class Classification of Known Objects

First, we focus on the part of the framework recognizing x in the case where it belongs to Y. We randomly select 10 samples from each $y \in Y$ and consider them as the test data, while the remaining 90 samples are used to train CNN_{XY}. We report in Table III the recognition accuracy that the framework can achieve when training data are available. We can see that the recognition accuracy is very high. This result is important because it has an impact on the training of $CONV_{XF}$ and thus also on recognizing novel objects. In addition, it reveals the efficiency of our framework in classifying objects when BioTacs data are available.

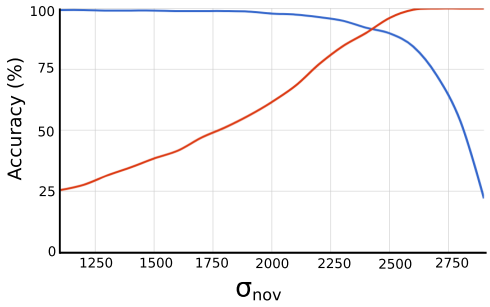


Fig. 7. Tuning σ_{nov}: the accuracy of classifying X_{te} as known (blue) and X_{val} as novel (red) for split 1.

TABLE II
ACCURACY OF NOVELTY DETECTION (%): DISTINCTION BETWEEN KNOWN AND NOVEL OBJECTS.

Split	$x \in X_{te}$	$x \in X_{val}$	$x \in X_Z$
1	89.8	96.2	94.7
2	91.0	95.5	89.7
3	78.1	86.7	93.3
4	92.7	96.8	81.3
5	89.2	95.2	94.7
6	96.7	98.5	83
7	94.8	98	67.5
average	90.3	95.3	89.5

TABLE III
RECOGNITION ACCURACIES (%) FOR MULTI-CLASS CLASSIFICATION OF Y WITH MANY TRAINING SAMPLES PER OBJECT.

Split	1	2	3	4	5	6	7	avg.
90 samples	96.2	95.1	90.8	96.2	95.9	96.5	95.4	95.2

D. Evaluation of Synthetic Features Generation

Synthetic features are generated in order to train the recognition system if real training data are missing. Thus, the quality of features generated using the algorithm presented in Sect. IV-C can be evaluated by the accuracy of novel object recognition after training the framework using synthetic features alone, and testing it on real features.

1) Comparison with real features : In table IV, we compare the recognition performance of test objects when real training features are available, and when they are replaced by synthetic training features. This was achieved by training FC_{FZ} once using real features extracted from BioTac data, and once with synthetic features generated according to GEN-F using the attribute vectors.

Several points emerge from Table IV:

TABLE IV
RECOGNITION ACCURACIES (%) FOR MULTI-CLASS CLASSIFICATION (REAL TRAINING DATA) AND ZSL (SYNTHETIC TRAINING DATA) WHEN TRAINING FC_{FZ} USING 0, 10, 50, 90 OR 100 SAMPLES PER CLASS.

Split	Training using real features				Training using synthetic features		
	0	10	50	90	10	50	100
1	17	88	97	98	36	34	35
2	17	95	100	100	24	22	23
3	17	95	95	100	20	10	10
4	17	97	98	100	36	38	37
5	17	68	85	100	32	34	33
6	17	70	88	100	33	34	33
7	17	67	77	83	35	31	33
average	17	83	91	97	31	29	29

TABLE V
RECOGNITION ACCURACIES (%) FOR ZSL USING GEN-F AND NEIGHB-F.

Split	1	2	3	4	5	6	7	avg.
GEN-F	36	24	20	**36**	32	33	**35**	31
NEIGHB-F	**37**	**33**	**37**	34	**41**	**34**	**35**	**36**

- It is obvious that the recognition accuracies are significantly higher after training with real features, than after training with synthetic features. This is important to highlight that ZSL cannot compete with multi-class classification, but replaces it when training data are not available. In fact, recognition using the BioTac readings is more efficient than using attributes. BioTacs give an average accuracy of 97% compared to 31% using attributes.
- The usefulness of ZSL can be observed from the performance of FC_{FZ} when no real data are available for any of the objects in Z (second column of the table). The multi-class classifier is not able to distinguish between objects in this case and classification accuracy is at chance level for 6 objects.
- On the other hand, for all object splits, ZSL could give a classification accuracy above chance. Generating synthetic features improved the recognition accuracy to 36% for splits 1 and 4, with an average accuracy of 31% for all splits.
- We notice that, contrary to real features, increasing the number of generated training samples does not necessarily improve the recognition. This is probably due to the fact that synthetic features of each class are generated from the same attribute vector (by the addition of a small amount of noise), and hence are similar. Due to this similarity, generating multiple features for an object probably leads to the over-fitting of the objects features in the training samples.

2) Zero-Shot Learning: Here, we analyze the recognition performance of novel objects, for which there is no real training data. For each x collected from a novel object, its feature vector is classified using FC_{FZ}, which was trained using generated features. We compare the classification performance of the two methods GEN-F and NEIGHB-F in Table V, using 10 generated training samples per class.

Results show that NEIGHB-F outperforms GEN-F for almost all splits, with an improvement of 5% of the average accuracy of all splits. NEIGHB-F reaches an accuracy of 41% in recognizing objects in Z of split 5, which is a considerable improvement compared to 17% obtained without generating the synthetic features.

TABLE VI
RECOGNITION ACCURACIES (%) FOR ZSL WITH GAN AND WITHOUT GAN.

Split	1	2	3	4	5	6	7	avg.
No-GAN	31	15	26	29	17	32	11	23
GAN	**37**	**33**	**37**	**34**	**41**	**34**	**35**	**36**

TABLE VII
RECOGNITION ACCURACIES (%) FOR ZSL WITH THE METHOD OF [38] AND NEIGHB-F.

Split	1	2	3	4	5	6	7	avg.
[38]	23	20	32	**48**	**43**	**52**	32	**36**
NEIGHB-F	**37**	**33**	**37**	34	41	34	**35**	**36**

Furthermore, to show the efficiency of our method, we compare it to other methods proposed for ZSL. First, we analyze the necessity of using the GAN-based setting, by skipping **Algorithm 2** and training the generator using **Algorithm 1** alone. This means that the generator is trained to generate synthetic features that are not necessarily similar to the real features. Results reported in table VI show a significant drop in performance when removing adversarial training from the learning algorithm. In fact, since FC_{FZ} is trained on generated features and tested on real ones, removing the GAN makes the generated features very different from the real ones.

Second, we compare our ZSL framework to the only previous study on haptic ZSL [38]. Table VII presents the comparison of recognition accuracies of all splits between the framework of [38] and NEIGHB-F. The latter performs better than [38] for 4 of the 7 splits. However, both methods have the same average accuracy. Therefore, for ZSL, the two methods perform quite similarly. Yet, the improvement we make w.r.t [38] is the possibility of recognizing previously trained and novel objects in the same framework. Furthermore, the current framework can integrate new training data for a smooth transition to multi-class classification, which were not possible in [38].

E. One-Shot Learning

To integrate a single training sample for each $z \in Z$, we use the method presented in Sect. IV-F to complete the training of FC_{FZ}, which was initially trained using NEIGHB-F (see Table V). Table VIII details the performance improvement achieved by the integration of a single training sample per class. We note that for most splits, performance is improved. For instance, adding one sample improves the accuracy of split 7 of 16%, and that of split 5 up to 55%. Overall, we obtain an average accuracy of 44% for all objects from all splits.

F. Recognition Performance of Split 1

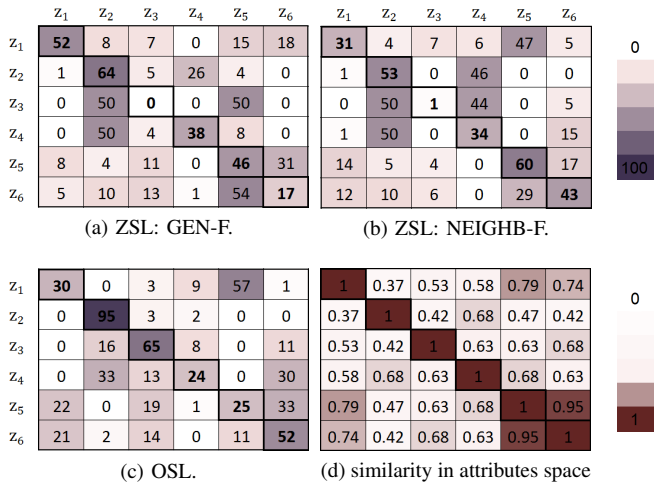
To further analyze the performance of our framework, we illustrate in Fig. 8a, 8b and 8c the confusion matrices of the

TABLE VIII
RECOGNITION ACCURACIES (%) FOR OSL.

Split	1	2	3	4	5	6	7	avg.
NEIGHB-F	37	33	**37**	**34**	41	34	35	36
OSL	**49**	**41**	**37**	28	**55**	**49**	**51**	**44**

TABLE IX
ZSL AND OSL RECOGNITION ACCURACIES OF 12 NOVEL OBJECTS.

Split	1	2	3	4	5	6	7
GEN-F	17	19	10	17	14	14	13
NEIGHB-F	24	19	18	13	14	13	15
OSL	31	30	24	27	31	27	34



(a) ZSL: GEN-F. (b) ZSL: NEIGHB-F.

(c) OSL. (d) similarity in attributes space

Fig. 8. Confusion matrices and attribute-based similarity matrix of split 1.

ZSL and the OSL classifications on split 1. In addition, Fig. 8d illustrates the similarity matrix of objects in Z by computing the Jaccard distance between the attribute vectors of each pair of objects. The most misclassified objects are confused with objects that are close to them in attributes space. This typically happens to object z_2, that is mostly misclassified as z_4, that is its most similar object according to the similarity matrix. This is not surprising since the training features have been generated from attributes, and close attributes vectors are expected to generate close features .

G. Robustness To the Number of Training Objects

Finally, we test the robustness of our ZSL and OSL recognition methods by reducing the number of training objects and increasing the number of novel ones. To this end, we randomly redefine new object splits such that Y contains 48 objects (80%) and Z contains the remaining 12 objects (20%). Table IX presents the recognition accuracies. As expected, the accuracies drop w.r.t. tables V and VIII, since we make the recognition task more challenging. However, crucially the recognition accuracies still remain above chance (8% for classifying 12 novel objects). These results provide some insights about the limit of our method to generalize novel objects, and shows that it cannot handle more challenging splits (like 70/30 or 60/40) especially with datasets of the size similar to PHAC2, which is unfortunately the largest tactile dataset available at present. However, these numbers can be arguably improved if the dataset is larger and more diverse.

VII. DISCUSSION AND CONCLUSION

In this work, we developed a recognition framework that is able to handle recognition of both known as well as novel objects. Our results showed the capacity of our framework to recognize objects having many training data (90 samples per class) with an average accuracy of 95% (Table III), in addition to recognizing 6 objects having no training data with an average accuracy of 36% (Table V), which was not possible using traditional training (Table IV). Furthermore, the framework efficiently integrates incoming data and reaches a high accuracy of multi-class classification when enough data becomes available with time (Table VIII for one sample, and Table IV for many samples). However, our framework still presents some limitations that can be a starting point for further improvements. First, recognition of novel objects is limited by the domain shift problem [49], and the correlation between attributes space and features space. Besides, the set of novel classes that our framework can recognize must be known, and adding novel classes requires the modification of the output layer of FC_{FZ}. Similarly, the addition of new attributes requires the modification of the input layer of G. This can however be solved by utilizing classifiers that add new classes with a low cost, as in [50]. Besides, our method has been tested only on the PHAC-2 dataset, and it may need to be adapted for use in other experimental setups.

One way to improve the current framework is to integrate visual data, when available. This is particularly promising, considering the good achievements of CNN and adversarial settings for image recognition and generation [48]. Visuo-tactile recognition can significantly improve recognition performance, since each modality can cope with problems faced by the other [43], [51]. In addition, we used the attributes designed by [42], which are semantic binary attributes. Exploring non-semantic and real-valued attributes can improve the accuracy and the generalization capability of the recognition [10], [23]. Finally, we recognize objects based on tactile data only, the current work can be extended by using vision as in [38], [52] However, even with these issues, our framework is probably the first tactile recognition system that handles objects with many, one, and zero training samples.

ACKNOWLEDGMENT

We would like to thank the authors of [42] for providing us with the PHAC-2 database. Zineb Abderrahmane is supported by the Ministry of Higher Education and Scientific Research of Algeria through the Excellence Fellowship.

REFERENCES

[1] A. Schmitz, Y. Bansho, K. Noda, H. Iwata, T. Ogata, and S. Sugano, "Tactile object recognition using deep learning and dropout," in *IEEE-RAS Int. Conf. on Humanoid Robots*, 2014, pp. 1044–1050.

[2] M. Madry, L. Bo, D. Kragic, and D. Fox, "ST-HMP: Unsupervised spatio-temporal feature learning for tactile data," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2014, pp. 2262–2269.

[3] H. Soh and Y. Demiris, "Incrementally learning objects by touch: Online discriminative and generative models for tactile-based recognition," *IEEE Trans. on Haptics*, vol. 7, no. 4, pp. 512–525, 2014.

[4] J. Hoelscher, J. Peters, and T. Hermans, "Evaluation of tactile feature extraction for interactive object recognition," in *IEEE-RAS Int. Conf. on Humanoid Robots*, 2015, pp. 310–317.

[5] J. Yang, H. Liu, F. Sun, and M. Gao, "Tactile sequence classification using joint kernel sparse coding," in *Int. Joint Conf. on Neural Networks (IJCNN)*, 2015, pp. 1–6.

[6] M. Kaboli, R. Walker, G. Cheng *et al.*, "In-hand object recognition via texture properties with robotic hands, artificial skin, and novel tactile descriptors," in *IEEE-RAS Int. Conf. on Humanoid Robots*, 2015, pp. 1155–1160.

[7] J. M. Gandarias, J. M. Gómez-de Gabriel, and A. García-Cerezo, "Human and object recognition with a high-resolution tactile sensor," in *SENSORS*. IEEE, 2017, pp. 1–3.

[8] H. Orii, S. Tsuji, T. Kouda, and T. Kohama, "Tactile texture recognition using convolutional neural networks for time-series data of pressure and 6-axis acceleration sensor," in *Int. Conf. on Industrial Technology (ICIT)*. IEEE, 2017, pp. 1076–1080.

[9] I. Biederman, "Recognition-by-components: a theory of human image understanding." *Psychological Review*, vol. 94, no. 2, p. 115, 1987.

[10] F. X. Yu, L. Cao, R. S. Feris, J. R. Smith, and S.-F. Chang, "Designing category-level attributes for discriminative visual recognition," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2013, pp. 771–778.

[11] C. H. Lampert, H. Nickisch, and S. Harmeling, "Learning to detect unseen object classes by between-class attribute transfer," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2009, pp. 951–958.

[12] M. Elhoseiny, B. Saleh, and A. Elgammal, "Write a classifier: Zero-shot learning using purely textual descriptions," in *IEEE Int. Conf. on Computer Vision (ICCV)*, 2013, pp. 2584–2591.

[13] P. Hensman and D. Masko, "The impact of imbalanced training data for convolutional neural networks," *Degree Project in Computer Science, KTH Royal Institute of Technology*, 2015.

[14] M. Buda, A. Maki, and M. A. Mazurowski, "A systematic study of the class imbalance problem in convolutional neural networks," *CoRR, abs/1710.05381*, 2017.

[15] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell, "Decaf: A deep convolutional activation feature for generic visual recognition," in *Int. Conf. on Machine Learning*, 2014, pp. 647–655.

[16] D. Xu, G. E. Loeb, and J. A. Fishel, "Tactile identification of objects using bayesian exploration," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2013, pp. 3056–3061.

[17] M. Kaboli and G. Cheng, "Novel tactile descriptors and a tactile transfer learning technique for active in-hand object recognition via texture properties," in *IEE-RAS Int. Conf. on Humanoid Robots - Workshop on Tactile sensing for manipulation: new progress and challenges*, 2016.

[18] M. Jin, H. Gu, S. Fan, Y. Zhang, and H. Liu, "Object shape recognition approach for sparse point clouds from tactile exploration," in *IEEE Int. Conf. on Robotics and Biomimetics (ROBIO)*, 2013, pp. 558–562.

[19] H. Liu, D. Guo, and F. Sun, "Object recognition using tactile measurements: Kernel sparse coding methods," *IEEE Trans. on Instrumentation and Measurement*, vol. 65, no. 3, pp. 656–665, 2016.

[20] H. Liu, Y. Wu, F. Sun, and D. Guo, "Recent progress on tactile object recognition," *Int. Journal of Advanced Robotic Systems*, vol. 14, no. 4, p. 1729881417717056, 2017.

[21] X. Yu and Y. Aloimonos, "Attribute-based transfer learning for object categorization with zero/one training example," *European Conf. on computer vision (ECCV)*, pp. 127–140, 2010.

[22] O. Russakovsky and L. Fei-Fei, "Attribute learning in large-scale datasets," in *European Conf. on computer vision (ECCV)*. Springer, 2010, pp. 1–14.

[23] D. Parikh and K. Grauman, "Relative attributes," in *IEEE Int. Conf. on Computer Vision (ICCV)*, 2011, pp. 503–510.

[24] P. Kankuekul, A. Kawewong, S. Tangruamsub, and O. Hasegawa, "Online incremental attribute-based zero-shot learning," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2012, pp. 3657–3664.

[25] D. Jayaraman and K. Grauman, "Zero-shot recognition with unreliable attributes," in *Advances in Neural Information Processing Systems*, 2014, pp. 3464–3472.

[26] Z. Al-Halah and R. Stiefelhagen, "How to transfer? zero-shot object recognition via hierarchical transfer of semantic attributes," in *IEEE Winter Conf. on Applications of Computer Vision (WACV)*, 2015, pp. 837–843.

[27] Y. Cheng, X. Qiao, X. Wang, and Q. Yu, "Random forest classifier for zero-shot learning based on relative attribute," *IEEE Trans. on Neural Networks and Learning Systems*, 2017.

[28] J. Lei Ba, K. Swersky, S. Fidler *et al.*, "Predicting deep zero-shot convolutional neural networks using textual descriptions," in *IEEE Int. Conf. on Computer Vision (ICCV)*, 2015, pp. 4247–4255.

[29] G. A. Miller, "Wordnet: a lexical database for english," *Communications of the ACM*, vol. 38, no. 11, pp. 39–41, 1995.

[30] M. Rohrbach, M. Stark, G. Szarvas, I. Gurevych, and B. Schiele, "What helps where–and why? semantic relatedness for knowledge transfer," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2010, pp. 910–917.

[31] T. Mensink, E. Gavves, and C. G. Snoek, "COSTA: Co-occurrence statistics for zero-shot classification," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2014, pp. 2441–2448.

[32] M. Rohrbach, M. Stark, and B. Schiele, "Evaluating knowledge transfer and zero-shot learning in a large-scale setting," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2011, pp. 1641–1648.

[33] S. Changpinyo, W.-L. Chao, and F. Sha, "Predicting visual exemplars of unseen classes for zero-shot learning," in *IEEE Int. Conf. on Computer Vision (ICCV)*, 2017, pp. 3496–3505.

[34] A. Mishra, M. Reddy, A. Mittal, and H. A. Murthy, "A generative model for zero shot learning using conditional variational autoencoders," *arXiv preprint arXiv:1709.00663*, 2017.

[35] Y. Long, L. Liu, F. Shen, L. Shao, and X. Li, "Zero-shot learning using synthesised unseen visual data with diffusion regularisation," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2017.

[36] Y. Xian, B. Schiele, and Z. Akata, "Zero-shot learning - the good, the bad and the ugly," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[37] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," in *Advances in Neural Information Processing Systems*, 2017, pp. 4080–4090.

[38] Z. Abderrahmane, G. Ganesh, A. Crosnier, and A. Cherubini, "Haptic Zero-Shot Learning: Recognition of objects never touched before," *Robotics and Autonomous Systems*, vol. 105, pp. 11–25, 2018.

[39] H. Liu, F. Sun, B. Fang, and D. Guo, "Cross-modal zero-shot-learning for tactile object recognition," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, pp. 1–9, 2018.

[40] M. Kaboli, R. Walker, and G. Cheng, "Re-using prior tactile experience by robotic hands to discriminate in-hand objects via texture properties," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2016, pp. 2242–2247.

[41] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[42] V. Chu, I. McMahon, L. Riano, C. G. McDonald, Q. He, J. M. Perez-Tejada, M. Arrigo, T. Darrell, and K. J. Kuchenbecker, "Robotic learning of haptic adjectives through physical interaction," *Robotics and Autonomous Systems (RAS)*, vol. 63, pp. 279–292, 2015.

[43] Y. Gao, L. A. Hendricks, K. J. Kuchenbecker, and T. Darrell, "Deep learning for tactile understanding from visual and haptic data," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2016, pp. 536–543.

[44] H. Liu, Y. Wu, F. Sun, D. Guo, and B. Fang, "Multi-label tactile property analysis," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2017, pp. 366–371.

[45] H. Liu, F. Sun, D. Guo, B. Fang, and Z. Peng, "Structured output-associated dictionary learning for haptic understanding," *IEEE Trans. on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 7, pp. 1564–1574, 2017.

[46] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Int. Conf. on Artificial Intelligence and Statistics*, 2010, pp. 249–256.

[47] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *ACM Int. Conf. on Multimedia*. ACM, 2014, pp. 675–678.

[48] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.

[49] Y. Fu, T. M. Hospedales, T. Xiang, and S. Gong, "Transductive multi-view zero-shot learning," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 37, no. 11, pp. 2332–2345, 2015.

[50] T. Mensink, J. Verbeek, F. Perronnin, and G. Csurka, "Distance-based image classification: Generalizing to new classes at near-zero cost," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 35, no. 11, pp. 2624–2637, 2013.

[51] H. Liu, Y. Yu, F. Sun, and J. Gu, "Visual–tactile fusion for object recognition," *IEEE Trans. on Automation Science and Engineering*, vol. 14, no. 2, pp. 996–1008, 2017.

[52] Z. Abderrahmane, G. Ganesh, A. Crosnier, and A. Cherubini, "Visuo-tactile recognition of daily-life objects never seen or touched before," 2018, iEEE Int. Conf. on Control Automation Robotics & Vision (ICARCV).