Generalization Error of Invariant Classifiers

Jure Sokolić[1] **Raja Giryes**[2] **Guillermo Sapiro**[3] **Miguel R. D. Rodrigues**[1]
[1]University College London [2]Tel-Aviv University [3]Duke University

Abstract

This paper studies the generalization error of invariant classifiers. In particular, we consider the common scenario where the classification task is invariant to certain transformations of the input, and that the classifier is constructed (or learned) to be invariant to these transformations. Our approach relies on factoring the input space into a product of a base space and a set of transformations. We show that whereas the generalization error of a non-invariant classifier is proportional to the complexity of the input space, the generalization error of an invariant classifier is proportional to the complexity of the base space. We also derive a set of sufficient conditions on the geometry of the base space and the set of transformations that ensure that the complexity of the base space is much smaller than the complexity of the input space. Our analysis applies to general classifiers such as convolutional neural networks. We demonstrate the implications of the developed theory for such classifiers with experiments on the MNIST and CIFAR-10 datasets.

1 Introduction

One of the fundamental topics in statistical learning theory is the one of the *generalization error* (GE). Given a training set and a hypothesis class, a learning algorithm chooses a hypothesis based on the training set in such a way that it minimizes an empirical loss. This loss, which is calculated on the training set, is also called the training loss and it often underestimates the expected loss. The GE is the difference between the empirical loss and the expected loss.

Proceedings of the 20^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2017, Fort Lauderdale, Florida, USA. JMLR: W&CP volume 54.

There are various approaches in the literature that aim at bounding the GE via the complexity measures of the hypothesis class, such as the VC-dimenension (Vapnik, 1999; Vapnik and Chervonenkis, 1991), the fat-shattering dimension (Alon et al., 1997), and the Rademacher and the Gaussian complexities (Bartlett and Mendelson, 2002). Another line of work provides the GE bounds based on the stability of the algorithms, by measuring how sensitive is the output to the removal or change of a single training sample (Bousquet and Elisseeff, 2002). Finally, there is a recent work by Xu and Mannor (2012) that bounds the GE in terms of the notion of algorithmic robustness.

An important property of the (traditional) GE bounds is that they are distribution agnostic, i.e., they hold for any distribution on the sample space. Moreover, GE bounds can lead to a principled derivation of learning algorithms with GE guarantees, e.g., Support Vector Machine (SVM) (Cortes and Vapnik, 1995) and its extension to non-linear classification with kernel machines (Hofmann et al., 2008).

However, the design of learning algorithms in practice does not rely only on the complexity measures of the hypothesis class, but it also relies on exploiting the underlying structure present in the data. A prominent example is associated with the field of computer vision where the features and learning algorithms are designed to be invariant to the intrinsic variability in the data (Soatto and Chiuso, 2016). Image classification is a particular computer vision task that requires representations that are invariant to various nuisances/transformations such as viewpoint and illumination variations commonly present in the set of natural images, but do not contain "helpful information" as to the identity of the classified object. This motivates us to develop a theory for learning algorithms that are invariant to certain sets of transformations.

The GE of invariant methods has been studied via the VC-dimension by Abu-Mostafa (1993), where it is shown that the subset of an hypothesis class that is invariant to certain transformations is smaller than the general hypothesis class. Therefore, it has a smaller VC-dimension. Yet, the authors do not pro-

vide any characterization of how much smaller the VC-dimension of an invariant method might be. Similarly, group symmetry in data distribution was also explored in the problem of covariance estimation, where it is shown that leveraging group symmetry leads to gains in sample complexity of the covariance matrix estimation (Shah and Chandrasekaran, 2012; Soloveychik et al., 2016).

There are various other examples in the literature that aims to understand/leverage the role of invariance in data processing. For example, Convolutional Neural Networks (CNNs) – which are known to achieve state of the art results in image recognition, speech recognition, and many other tasks (LeCun et al., 2015) – are known to possess certain invariances. The invariance in CNNs is achieved by careful design of the architecture so that it is (approximately) invariant to various transformations such as rotation, scale and affine deformations (Cohen and Welling, 2016; Dieleman et al., 2016; Gens and Domingos, 2014); or by training with augmented training set, meaning the training set is augmented with some transformed versions of the training samples, so that the learned network is approximately invariant (Simard et al., 2003). Another example of a translation invariant method is the scattering transform, which is a CNN-like transform based on wavelets and point-wise non-linearities (Bruna and Mallat, 2012). See also (Sifre and Mallat, 2013; Wiatowski and Bölcskei, 2015). In practice, such learning techniques achieve a lower GE than their "non-invariant" counterparts.

Poggio et al. (2012) and Anselmi et al. (2014, 2016) study biologically plausible learning of invariant representations and connect their results to CNNs. The role of convolutions and pooling in the context of natural images is also studied by Cohen and Shashua (2016).

There are various works that that study the GE of CNNs (Huang et al., 2015; Neyshabur et al., 2015; Shalev-Shwartz and Ben-David, 2014; Sokolić et al., 2016), however, they do not establish any connection between the network's invariance and its GE.

Motivated by the above examples, this work proposes a theoretical framework to study the GE of invariant learning algorithms and shows that an invariant learning technique may have a much smaller GE than a non-invariant learning technique. Moreover, our work directly relates the difference in GE bounds to the size of the set of transformations that a learning algorithm is invariant to. Our approach is significantly different from (Abu-Mostafa, 1993) because it focuses on the complexity of the data, rather than on the complexity of the hypothesis class.

1.1 Contributions

The main contribution of this paper can be summarized as follows:

We prove that given a learning method invariant to a set of transformations of size T, the GE of this method may be up to a factor $\sqrt{T}$ smaller than the GE of a non-invariant learning method.

Additionally, our other contributions include:

- We define notions of stable invariant classifiers and provide GE bounds for such classifiers;
- We establish a set of sufficient conditions that ensure that the bound of the GE of a stable invariant classifier is much smaller than the GE of a robust non-invariant classifier. We are not aware of any other works in the literature that achieve this;
- Our theory also suggests that explicitly enforcing invariance when training the networks should improve the generalization of the learning algorithm. The theoretical results are supported by experiments on the MNIST and CIFAR-10 datasets.

2 Problem Statement

We start by describing the problem of supervised learning and its associated GE. Then we define the notions of invariance in the classification task and the notion of an invariant algorithm.

2.1 Generalization Error

We consider learning a classifier from training samples. In particular, we assume that there is a probability distribution P defined on the sample space $\mathcal{Z}$ and that we have a training set drawn i.i.d. from P denoted by $S_m = \{s_i\}_{i=1}^m$, $s_i \in \mathcal{Z}$, $i = 1, \ldots, m$. A learning algorithm $\mathcal{A}$ takes the training set S_m and maps it to a learned hypothesis $\mathcal{A}_{S_m}$. The loss function of an hypothesis $\mathcal{A}_{S_m}$ on the sample $z \in \mathcal{Z}$ is denoted by $l(\mathcal{A}_{S_m}, z)$. The empirical loss and the expected loss of the learned hypothesis $\mathcal{A}_{S_m}$ are defined as

$$l_{\text{emp}}(\mathcal{A}_{S_m}) = 1/m \sum_{s_i \in S_m} l\left(\mathcal{A}_{S_m}, s_i\right) \quad \text{and} \tag{1}$$

$$l_{\text{exp}}(\mathcal{A}_{S_m}) = \mathbb{E}_{s\sim P}\left[l\left(\mathcal{A}_{S_m}, s\right)\right], \tag{2}$$

respectively; and the GE is defined as

$$GE(\mathcal{A}_{S_m}) = \left|l_{\text{emp}}(\mathcal{A}_{S_m}) - l_{\text{exp}}(\mathcal{A}_{S_m})\right|. \tag{3}$$

We consider a classification problem, where the sample space $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$ is a product of the input space

$\mathcal{X}$ and the label space $\mathcal{Y}$, where a vector $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^N$ represents an observation that has a corresponding class label $y \in \mathcal{Y} = \{1, 2, \ldots, N_\mathcal{Y}\}$. We will write $z = (\mathbf{x}, y)$ and $s_i = (\mathbf{x}_i, y_i)$.

2.2 Stable Classifier and its Generalization

The feature extractor (e.g., CNN) used in this work defines the non-linear function $f(\mathbf{x}, \theta) : \mathbb{R}^N \to \mathbb{R}^{N_\mathcal{Y}}$, where $N_\mathcal{Y}$ represents the number of classes, N represents the dimension of the input signal, and θ represents the parameters of the feature extractor. The classifier defined by the feature extractor is then given as

$$\arg\max_{i \in [N_\mathcal{Y}]} (f(\mathbf{x}, \theta))_i \,, \tag{4}$$

where, $(f(\mathbf{x}, \theta))_i$ is the i-th element of $f(\mathbf{x}, \theta)$. For example, this may correspond to a CNN with a softmax layer at the end. We will often write $f(\mathbf{x}, \theta) = f(\mathbf{x})$, and define its Jacobian matrix as

$$\mathbf{J}(\mathbf{x}, \theta) = \frac{df(\mathbf{x}, \theta)}{d\mathbf{x}} = \mathbf{J}(\mathbf{x}) \,. \tag{5}$$

A learning algorithm $\mathcal{A}$ therefore returns a hypothesis, which is a function of the training set S_m,

$$\mathcal{A}_{S_m}(\mathbf{x}) = \arg\max_{i \in [N_\mathcal{Y}]} (f(\mathbf{x}, \theta(S_m)))_i \,. \tag{6}$$

In a classification task, the goal of learning is to find an hypothesis that separates training samples from different classes. To model this we define the score of a training sample, which measures how confident the prediction of a classifier is:

Definition 1 (Score)**.** *Consider a training sample $s_i = (\mathbf{x}_i, y_i)$. The score of training sample s_i is defined as*

$$o(s_i) = \min_{j \neq y_i} \sqrt{2} \left((f(\mathbf{x}_i))_{y_i} - (f(\mathbf{x}_i))_j \right) \,. \tag{7}$$

Note that a large score of training samples does not imply that the learned hypothesis will have a small GE. In this work we leverage the (non-invariant) GE bounds provided by Sokolić et al. (2016). Before providing such bounds we define the notion of learning algorithm stability and the notion of covering number that are crucial for the GE bounds.

Definition 2 (Stable learning algorithm)**.** *Consider the algorithm $\mathcal{A}$ and the hypothesis $\mathcal{A}_{S_m}(\mathbf{x})$ given in* (6)*. The learning algorithm $\mathcal{A}$ is stable if for any training set S_m*

$$\max_{\mathbf{x} \in \mathbb{R}^N} \|\mathbf{J}(\mathbf{x})\|_2 \leq 1 \,, \tag{8}$$

where $\|\cdot\|_2$ denotes the spectral norm.

Stability of a learning algorithm defined in this way ensures that a learned classifier has a small GE as we shall see in Theorem 1.

We also need a measure of complexity/size of the input space $\mathcal{X}$, which is given by the covering number.

Definition 3. *Consider a space $\mathcal{X}$ and a metric d. We say that the set $\mathcal{C}$ is an ϵ-cover of $\mathcal{X}$ if $\forall \mathbf{x} \in \mathcal{X}$, $\exists \mathbf{x}' \in \mathcal{C}$ such that $d(\mathbf{x}, \mathbf{x}') \leq \epsilon$. The covering number of $\mathcal{X}$ corresponds to the cardinality of the smallest $\mathcal{C}$ that covers $\mathcal{X}$. It is denoted by $\mathcal{N}(\mathcal{X}; d, \epsilon)$.*

In this work we will assume that d is the Euclidean metric: $d(\mathbf{x}, \mathbf{x}') = \|\mathbf{x} - \mathbf{x}'\|_2$.

Finally, we can provide the GE bounds for the stable learning algorithm. This is a variation of theorems 2 and 4 by Sokolić et al. (2016).[1]

Theorem 1. *Assume that the learning algorithm $\mathcal{A}$ is stable and that there exists a constant γ such that*

$$o(s_i) \geq \gamma \quad \forall s_i \in S_m \,. \tag{9}$$

Assume also that the loss $l(\cdot)$ is the 0-1 loss. Then, for any $\delta > 0$, with probability at least $1 - \delta$,

$$GE(\mathcal{A}_{S_m}) \leq \sqrt{\frac{2\log(2) \cdot N_\mathcal{Y} \cdot \mathcal{N}(\mathcal{X}; d, \gamma/2)}{m}} + \sqrt{\frac{2\log(1/\delta)}{m}} \,. \tag{10}$$

Proof. The proof is straightforward by the application of theorems 2 and 4 by Sokolić et al. (2016). □

The GE therefore approaches zero with rate $1/\sqrt{m}$ and it depends on the number of classes via $\sqrt{N_\mathcal{Y}}$. Critical is the dependence on the covering number $\mathcal{N}(\mathcal{X}; d, \gamma/2)$, which is a function of the input space $\mathcal{X}$ and the margin γ.

2.3 Structured Input Space and Invariant Algorithms

The bound of the GE provided in the previous section depends on the covering of the input space $\mathcal{X}$. As noted in the introduction, $\mathcal{X}$ often exhibits symmetries that may reduce its "effective" complexity and therefore also reduce the GE. We formalize this intuition in this section.

To capture the additional structure present in the data, we model the input space $\mathcal{X}$ as a product of a base space $\mathcal{X}_0$ and a set of transformations $\mathcal{T}$:

$$\mathcal{X} = \mathcal{T} \times \mathcal{X}_0 \coloneqq \{t(\mathbf{x}) : t \in \mathcal{T}, \mathbf{x} \in \mathcal{X}_0\} \,, \tag{11}$$

[1] Sokolić et al. (2016) also provide tighter GE bounds. For the sake of simplicity we use the bounds based on the spectral norm of the Jacobian matrix.

where $\mathcal{X}_0 \subseteq \mathbb{R}^N$, $\mathcal{T} = \{t_0, t_2, \dots t_{T-1}\}$ and T corresponds to the size of $\mathcal{T}$.[2] We assume t_0 to be the identity, i.e., $t_0(\mathbf{x}) = \mathbf{x}$ throughout this work. For example, if $\mathcal{X}_0$ is a set of images and $\mathcal{T}$ is a set of translations, then $\mathcal{X}$ will be the set of images with all possible translations of the images in $\mathcal{X}_0$. See also Figure 1.

We assume that the classification task is invariant to the set of transformations $\mathcal{T}$, i.e., we are really interested only in the set $\mathcal{X}_0$ but have access to transformed samples of it, where clearly all of them have the same label. In other words, the class labels of $t(\mathbf{x})$ are the same for all $t \in \mathcal{T}$. In this case, it is reasonable to leverage this by using an invariant learning algorithm.[3]

Definition 4 (Invariant algorithm)**.** *A learning algorithm $\mathcal{A}$ is invariant to the set of transformations $\mathcal{T}$ if the embedding is invariant:*

$$f(t_i(\mathbf{x}), S_m) = f(t_j(\mathbf{x}), S_m) \quad \forall \mathbf{x} \in \mathcal{X}_0, t_i, t_j \in \mathcal{T}\,, \tag{12}$$

for any training set S_m. We will denote such learning algorithm by $\mathcal{A}^{\mathcal{T}}_{S_m}$.

This leads us to the question that will occupy us throughout this paper: what is the GE of an invariant learning algorithm.

3 Generalization Error of Invariant Classifiers

In this section we provide bounds to the GE of invariant algorithms. The invariance of the learning method induces a possibly more efficient covering of the input space $\mathcal{X}$, which translates into a lower GE.

The GE of invariant and stable learning algorithms can be bounded as follows:

Theorem 2. *Assume that the learning algorithm $\mathcal{A}$ is stable and invariant to $\mathcal{T}$ and that there exists a constant γ such that*

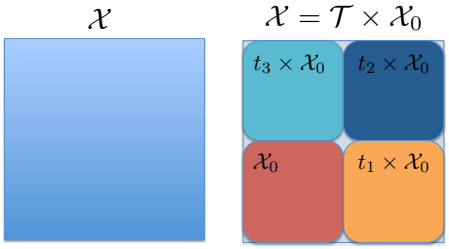
$$o(s_i) \geq \gamma \quad \forall s_i \in S_m\,. \tag{13}$$

Assume also that the loss $l(\cdot)$ is the 0-1 loss. Then, for any $\delta > 0$, with probability at least $1 - \delta$,

$$GE(\mathcal{A}^{\mathcal{T}}_{S_m}) \leq \sqrt{\frac{2\log(2)\cdot N_{\mathcal{Y}} \cdot \mathcal{N}(\mathcal{X}_0; d, \gamma/2)}{m}} + \sqrt{\frac{2\log(1/\delta)}{m}}\,. \tag{14}$$

[2]Note that the discrete representation of this set is not limiting in practice.

[3]Here we define a notion of absolute invariance. It is easy to extend it to approximate invariance, where in $\mathcal{X}$ we have transformed versions of $\mathcal{X}_0$ plus small/bounded noise; and also to extend the GE bounds in a similar manner for approximately invariant learning algorithms.



(a) Input space. (b) Input space decomposition.

Figure 1: Theorem 1 shows that the size of the input space $\mathcal{X}$ determines the GE of a stable learning algorithm. The input space can often be constructed as a product of a simpler base space $\mathcal{X}_0$ and a set of transformations $\mathcal{T}$, where the transformations in $\mathcal{T}$ preserve the class labels. Theorem 2 shows that the GE of an invariant stable learning algorithm is determined by the size of the base space $\mathcal{X}_0$. The size of the base space $\mathcal{X}_0$ can be much smaller than the size of the input space $\mathcal{X}$.

Proof. We show that under the assumptions of this theorem the learning algorithms is $(\mathcal{N}(\mathcal{X}_0; d, \gamma/2), 0)$-robust (see (Xu and Mannor, 2012) or (Sokolić et al., 2016)). The GE bound then follows from Theorem 3 and Example 9 by Xu and Mannor (2012) (or theorems 1 and 2 by Sokolić et al. (2016)).

We construct a covering as follows. Take the covering that leads to the covering number $\mathcal{N}(\mathcal{X}_0; d, \gamma/2)$ and denote the subsets of $\mathcal{X}_0$ by $\mathcal{K}_i$, $i = 1, \dots, \mathcal{N}(\mathcal{X}_0; d, \gamma/2)$. By the definition of $\mathcal{X}$ in (11) we can cover $\mathcal{X}$ by $\mathcal{N}(\mathcal{X}_0; d, \gamma/2)$ sets of the form $\mathcal{T} \times \mathcal{K}_i$, $i = 1, \dots, \mathcal{N}(\mathcal{X}_0; d, \gamma/2)$.

Now take $\mathbf{x}_i$ in the training set and $\mathbf{x} \in \mathcal{X}$ such that $\mathbf{x}_i, \mathbf{x} \in \mathcal{T} \times \mathcal{K}_j$. Due to the invariance of f we have $\|f(\mathbf{x}_i) - f(\mathbf{x})\|_2 < \gamma$ and all $\mathbf{x}$ will lie in the same decision region as $\mathbf{x}_i$. This implies that stable and invariant learning algorithm is $(\mathcal{N}(\mathcal{X}_0; d, \gamma/2), 0)$-robust. The GE bound follows from Theorem 3 by Xu and Mannor (2012). □

Note that the GE bound in Theorem 2 is of the same form as the GE bound in Theorem 1 and the main difference is in the employed covering number. In particular, the ratio between the bounds is

$$R(\mathcal{X}_0, \mathcal{X}; d, \epsilon) = \left(\frac{\mathcal{N}(\mathcal{X}_0; d, \epsilon)}{\mathcal{N}(\mathcal{X}; d, \epsilon)}\right)^{1/2}\,, \tag{15}$$

where $\epsilon = \gamma/2$ in our case. We are especially interested in the scenarios where the GE bound of an invariant method is much smaller than the GE

bound of a non-invariant method. This happens when $R(\mathcal{X}_0, \mathcal{X}; d, \epsilon) \ll 1$. We now establish a set of sufficient conditions on $\mathcal{X}_0$, $\mathcal{T}$, d and ϵ such that $R(\mathcal{X}_0, \mathcal{X}; d, \epsilon) \ll 1$.

Theorem 3. *Assume that $\mathcal{X} = \mathcal{T} \times \mathcal{X}_0$ and choose $\epsilon < 1$. Then*

$$d(t(\mathbf{x}), t'(\mathbf{x}')) > 2\epsilon \quad \forall \mathbf{x}, \mathbf{x}' \in \mathcal{X}_0, t \neq t' \in \mathcal{T} \quad (16)$$

and

$$d(t(\mathbf{x}), t(\mathbf{x}')) \geq d(\mathbf{x}, \mathbf{x}') \quad \forall \mathbf{x}, \mathbf{x}' \in \mathcal{X}_0, t \in \mathcal{T} \quad (17)$$

$$\implies R(\mathcal{X}_0, \mathcal{X}; d, \epsilon) \leq 1/\sqrt{T}, \quad (18)$$

where T is the number of elements in $\mathcal{T}$. On the other hand,

$$d(t(\mathbf{x}), t'(\mathbf{x})) = 0 \quad \forall \mathbf{x} \in \mathcal{X}_0, t \neq t' \in \mathcal{T} \quad (19)$$

$$\implies R(\mathcal{X}_0, \mathcal{X}; d, \epsilon) = 1. \quad (20)$$

Proof. Consider any covering of $\mathcal{X}_0$ that leads to the covering number $\mathcal{N}(\mathcal{X}_0; d, \epsilon)$. Denote the metric balls of radius ϵ that cover $\mathcal{X}_0$ by $\mathcal{C}_i$, $i = 1, \ldots, \mathcal{N}(\mathcal{X}_0; d, \epsilon)$. Denote the elements of $\mathcal{T}$ as t_j, $j = 1, \ldots, T$ and the transformed sets by $t_j(\mathcal{X}_0) = \{t_j(\mathbf{x}) : \mathbf{x} \in \mathcal{X}_0\}$, $j = 1, \ldots, T$.

First, we show that (16) implies that any possible metric ball of radius ϵ can only have non-empty intersection with one of the "copies" of $\mathcal{X}_0$. Denote by $\mathcal{B}$ an arbitrary metric ball of radius ϵ. Then

$$\mathcal{B} \cap t_j(\mathcal{X}_0) \neq \emptyset \implies \mathcal{B} \cap t_k(\mathcal{X}_0) = \emptyset \quad \forall k \neq j. \quad (21)$$

To see this, observe that the definition of $\mathcal{B}$ implies that $d(\mathbf{x}, \mathbf{x}') \leq 2\epsilon$, $\forall \mathbf{x}, \mathbf{x}' \in \mathcal{B}$. Now take a point $\mathbf{x} \in \mathcal{B} \cap t_j(\mathcal{X}_0)$ and a point $\mathbf{x}' \in t_k(\mathcal{X}_0)$, $k \neq j$. Note that by (16) $d(\mathbf{x}, \mathbf{x}') > 2\epsilon$, which implies that $\mathbf{x}' \notin \mathcal{B}$ and therefore $\mathcal{B} \cap t_k(\mathcal{X}_0) = \emptyset$. This implies that the covering number of $\mathcal{X}$ with metric ball of radius ϵ is

$$\mathcal{N}(\mathcal{X}; d, \epsilon) = \sum_{j=1}^{T} \mathcal{N}(t_j(\mathcal{X}_0); d, \epsilon). \quad (22)$$

Finally, it remains to be proven that $\mathcal{N}(t_j(\mathcal{X}_0); d, \epsilon) \geq \mathcal{N}(\mathcal{X}_0; d, \epsilon)$ $\forall t_j \in \mathcal{T}$, which is straightforward to establish given the condition (17). This proves (18). Proof of (20) is trivial as $\mathcal{X}_0 = \mathcal{X}$ when (19) holds. □

We have shown, via conditions on the geometry of the base space $\mathcal{X}_0$, and the effect of transformations in $\mathcal{T}$ on it, that the ratio $R(\mathcal{X}_0, \mathcal{X}; d, \epsilon)$ can be smaller or equal to $1/\sqrt{T}$. Note that conditions (16) and (17) ensure that the effect of transformations in $\mathcal{T}$ can not be captured by the metric d. Otherwise, the invariant algorithm has no advantage over a non-invariant one (this is illustrated by examples in Section 3.1):

- The sufficient condition in (16) can be stated as follows. Take any pair of vectors in the base space $\mathcal{X}_0$ and transform them by the two transformations in $\mathcal{T}$ that are not equal. Then the distance between the pair of vectors must be at least 2ϵ. In other words, the transformation must not make two distinct vectors in the base space $\mathcal{X}_0$ (distance at least 2ϵ) indistinguishable (distance smaller than 2ϵ). Similarly, any two transformations in $\mathcal{T}$ that are not equal must make two similar vectors in $\mathcal{X}_0$ (with distance smaller than 2ϵ) distinct (distance at least 2ϵ).
- The sufficient condition in (17) ensures that the transformations in $\mathcal{T}$ are not trivial, i.e., they do not reduce the complexity of the base space $\mathcal{X}_0$. For example, a transformation that maps any $\mathbf{x} \in \mathcal{X}_0$ into itself violates (17) and leads to a set of the same complexity, as formalized by (19).

The results of this section can be summarized by the following remark:

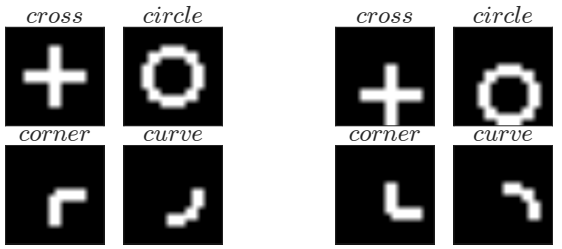
Remark 1. *Given an input space $\mathcal{X}$, which is structured according to the assumptions of Theorem 3 and the size of transformation set T, we have established that the GE of an invariant stable learning algorithm may be up to a factor $\sqrt{T}$ smaller than the GE of a non-invariant stable learning algorithm. To the best of our knowledge, this is the first time such quantitative result is provided for invariant algorithms.*

3.1 Illustration

To provide additional intuition related to Theorem 3, we present the following toy example. We consider four images of dimension $N \times N$, with $N = 16$, Figure 2(a). The sets of transformations that we consider are:

- Translation set: The set of pixel-wise cyclic translations in any direction. The size of the set is N^2.
- Rotation set: The set of image rotations by 90°. The size of this set is 4 (this may explain why the 90° rotation invariance is useful but not as critical as the translation invariance).
- Trans-rotation set: A product of the translation and the rotation sets, where the rotation is applied first followed by a translation. The size of this set is $4 \times N^2$.

Note that all the transformations above can be implemented by permutation matrices which are orthonormal. This is important as it implies that all the considered sets satisfy the condition in (17). Examples of transformed atoms are shown in Figure 2(b).



(a) Atoms. (b) Transformed atoms.

Figure 2: (a) A set of atoms (*cross*, *circle*, *corner*, *curve*) used to construct the base space. (b) Examples of transformed atoms with a transformation from the trans-rotation set.

We now provide an example of a base space $\mathcal{X}_0$ and a transformation set $\mathcal{T}$ for which $R(\mathcal{X}_0, \mathcal{X}; d, \epsilon) \leq 1/\sqrt{T}$; and then provide an example of a base space $\mathcal{X}_0$ and a transformation set $\mathcal{T}$ for which $R(\mathcal{X}_0, \mathcal{X}; d, \epsilon) \not\leq 1/\sqrt{T}$.

Example for $R(\mathcal{X}_0, \mathcal{X}; d, \epsilon) \leq 1/\sqrt{T}$: Consider $\mathcal{X}_0 = \{cross, circle, corner, curve\}$ and $\mathcal{T}$ to be the translation set. The set $\mathcal{X} = \mathcal{T} \times \mathcal{X}_0$ then contains all possible translations of shapes in $\mathcal{X}_0$. We have verified numerically that the condition in (16) is satisfied for all $\epsilon < 0.375$. Therefore, $R(\mathcal{X}_0, \mathcal{X}; d, \epsilon) \leq 1/\sqrt{T}$ for $\epsilon < 0.375$, where $\sqrt{T} = N = 16$ is the dimension of the images. Therefore, a translation invariant learning method can attain a GE with a factor N smaller than the GE of a non-invariant method.

Similarly, if we take $\mathcal{X}_0 = \{corner, curve\}$ and $\mathcal{T}$ to be the trans-rotation set, we can establish $R(\mathcal{X}_0, \mathcal{X}; d, \epsilon) \leq 1/(2N)$ for $\epsilon < 0.26$.

Examples for $R(\mathcal{X}_0, \mathcal{X}; d, \epsilon) \not\leq 1/\sqrt{T}$: Now consider $\mathcal{X}_0 = \{cross, circle\}$ and $\mathcal{T}$ to be the rotation set. Therefore, $\mathcal{X} = \mathcal{T} \times \mathcal{X}_0$ contains all possible 90° rotations of *circle* and *cross* in Figure 2(a). It is clear that the *circle* and *cross* are already invariant to such rotation, i.e., they corresponds to exactly the same shape. Therefore, the condition in (19) holds and $R(\mathcal{X}_0, \mathcal{X}; d, \epsilon) = 1$. Clearly, in such cases, an invariant learning algorithm is not expected to have a smaller GE than a non-invariant learning algorithm.

4 Invariant CNNs

In this section we discuss the implication of our theory on CNNs, which are very popular for classification. Note that this is one particular example and that our theory holds also for other possible classifiers. We consider two ways for which invariance can be achieved for CNNs: via an appropriate construction of the CNN architecture or by training them to be invariant.

Invariance of the CNN Architecture- Given that the set of transformations is a group, averaging a function over a group leads to an invariant representation (Anselmi et al., 2014; Bruna and Mallat, 2012; Cohen and Welling, 2016). For example, in conventional CNNs the pooling operators usually average over the translation group and make the CNNs translation invariant.

Cohen and Welling (2016) generalize the notion of convolution over translation group to general groups, which leads to architectures that can be invariant to arbitrary transformations that form discrete groups. A related approach involves normalization of the network input, which eliminates the effect of affine transformations of the input (Jaderberg et al., 2015).

Invariance of the CNN Learning- As an alternative to encoding the invariances in the CNN architecture we can train a CNN to become invariant. This is particularly helpful in the cases that we do not know exactly how to characterize or impose the invariance manually on the network. Such an "approximate invariance" is achieved by training CNNs with data augmentation, which involves training the network with the transformed samples of the training examples. This was indicated by Lenc and Vedaldi (2015), who showed that CNNs trained on the ImageNet *implicitly* learn to be invariant to flips, scalings and rotations.

Our theory suggests that enforcing the invariance of the CNN representation *explicitly* should improve the robustness of CNNs and improve their GE. For example, we may train networks with an explicit regularization term of the form

$$\sum_{t \in \mathcal{T}} \|f(\mathbf{x}_i) - f(t(\mathbf{x}_i))\|_2^2 \,, \tag{23}$$

which promotes the invariance of the representation. We validate the effectiveness of this regularization in Section 5.2.

5 Experiments

We now demonstrate the theoretical results with experiments on the MNIST and CIFAR-10 datasets.

5.1 Rotation Invariant CNN

Here we compare a rotation invariant CNN and a conventional CNN on rotated MNIST datasetses. The rotated MNIST-D° dataset is constructed by rotating the digits by an angle $r \cdot D^\circ$, $r \in \{0, 1, 2, \ldots, 360/D - 1\}$, where the index r is chosen randomly for each image in the dataset. We use $D = 180, 90, 45$.

We use a 7 layer CNN architecture: $(32, 5, 5)$-conv, $(2, 2)$-max-pool, $(64, 5, 5)$-conv, $(2, 2)$-max-pool, $(128, 5, 5)$-conv followed by a global average pooling layer and a softmax layer, where (k, u, v)-conv denotes the convolutional layer with k filters of size $u \times v$, and (p, p)-max-pool denotes the max-pooling layer with pooling regions of size $p \times p$. The rotation invariant CNN is the same as the conventional CNN, but it includes a cyclic slice layer before the first convolutional layer and a cyclic pool layer before the softmax layer. Both, the cyclic slice layer and the cyclic pool layer were proposed by Dieleman et al. (2016) and together they ensure that the CNN is invariant to rotations. In particular, the cyclic slice layer takes input image $\mathbf{x}$ and creates copies of $\mathbf{x}$, each rotated for $r \cdot D^\circ$, $r = 0, 1, 2, \ldots, 360/D - 1$, where D is the same as in the dataset MNIST-D°. The copies are then passed through the CNN independently. At the end of the CNN, before the softmax layer, the outputs of the copies are averaged by a cyclic pool layer to obtain a rotation invariant representation.

The networks are trained using stochastic gradient descent (SGD) with momentum, which was set to 0.9. The training objective is the standard categorical cross entropy (CCE) loss. Batch size was set to 32 and learning rate was set to 0.01 and reduced by 10 after 100 epochs. The networks were trained for 150 epochs in total. Weight decay regularization was set to 10^{-4}. We used training sets of sizes 10^3, 10^4, $2 \cdot 10^4$, $5 \cdot 10^4$.

The classification accuracies are reported in Figure 3(a), the GE is reported in Figure 3(b) and the ratio of the GEs of the invariant and the conventional CNNs are shown in Figure 3(c). We may note that the (explicitly) rotation invariant CNN always has a higher classification accuracy than the conventional CNN. Moreover, the GE of the rotation invariant CNN is much smaller than the GE of the conventional CNN. The difference is most significant when the training set is small, which demonstrates the importance of invariance for the generalization of learning algorithms.

Note also that the GE of the rotation invariant CNNs on different datasets MNIST-D°, $D = 180, 90, 45$, is roughly the same, whereas the conventional CNNs have a higher GE on the datasets with a smaller D. This can be explained by the fact that the rotated MNIST dataset with a smaller D is more complex due to the larger number of rotations. The sizes of the transformation sets for $D = 180, 90, 45$ are $2, 4$ and 8, respectively. Theorem 3 predicts that the ratio of the GEs of an invariant and a non-invariant CNNs is equal to $\sqrt{|\mathcal{T}|}$. The actual ratios are shown in Figure 3(c). We can observe that the GE ratios obtained empirically roughly follow the theoretical prediction. However, when the training set is small, the conventional CNN generalizes worse than predicted by our theory and when the training set is large, the conventional CNN generalizes better than predicted by our theory. We conjecture that the conventional CNNs learn to be "partially" invariant when the number of training samples is large. Moreover, the current theory might not capture the relationship between invariant and non-invariant CNNs entirely, especially when the assumptions of Theorem 3 do not hold.

Finally, we also consider the rotation invariant MNIST dataset, where each image $\mathbf{x}$ in the dataset is rotated by $r \cdot D^\circ$, $r \in \{0, 1, 2, \ldots, 360/D - 1\}$ and the $360/D$ copies are averaged to obtain a sample. As our theory suggests, the rotation invariant CNNs in this case do not have a lower GE than a conventional CNN because the dataset itself is rotation invariant. In fact, given the rotation invariant MNIST dataset, the rotation invariant CNN and the conventional CNN are equivalent. This can be easily established by observing that the cyclic slicing layer produces copies of the input that are identical. We have verified empirically that the rotation invariant and the non-invariant CNNs perform the same on the rotation invariant MNIST dataset.

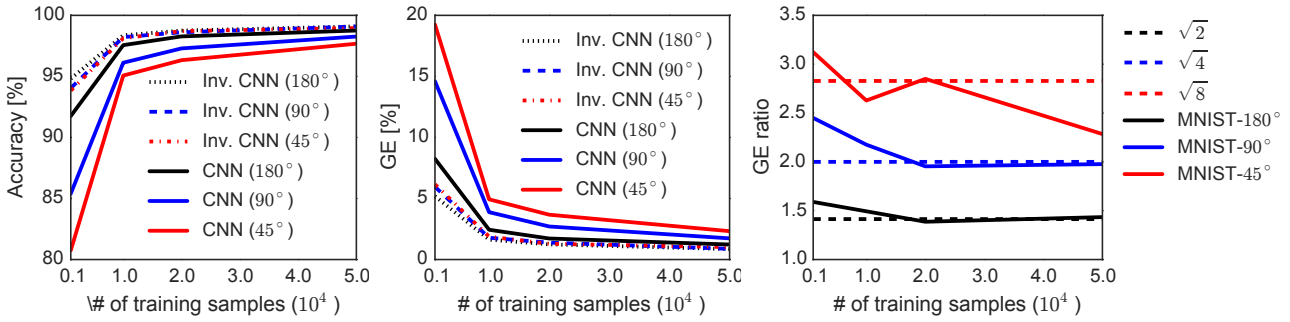
5.2 Learning the Invariances

Finally, we demonstrate that learning invariances explicitly can lead to a lower GE. We use the CIFAR-10 dataset, which is normalized following (Zagoruyko and Komodakis, 2016), and the Wide ResNet (Zagoruyko and Komodakis, 2016) with 13 layers of width 5.

The networks are trained using SGD and the learning rate is set to 0.01 for the first epoch and then to 0.05, 0.005 and 0.0005, each for 30 epochs. We use 10^3, 10^4, $2 \cdot 10^4$ and $5 \cdot 10^4$ training samples. We have found that using the Jacobian regularization (Sokolić et al., 2016) improves performance in all cases and it's factor is set to 0.1 with smaller training sets (2500, 5000, 10000) and 0.05 otherwise. Batch size is set to 128.

SGD batches are constructed as follows: the first half of the batch contains images from the training set and the other half of the mini batch contains transformed versions of the images in the first half of the mini batch where the transformations are chosen at random. The set of transformations contains shifts of ± 4 pixels and horizontal flips, as in (Zagoruyko and Komodakis, 2016).

We promote the invariance by using the regularizer in (23). We chose to regularize the output of the last global pooling layer instead of the softmax output and use the corresponding pairs from the batch to compute (23). The regularization factor in all experiments is set to 10^{-4}.



(a) Test set classification acc. (b) Generalization error. (c) Generalization error ratio.

Figure 3: (a) Clasification accuracy, (b) the GE of the rot. invariant CNN and the conventional CNN and (c) the ratio of the GEs of the rotation invariant CNN and the conventional CNN on the rotated MNIST datasets.

Table 1 reports the standard test accuracy and the accuracy of the predictions averaged over the augmented test set (denoted by + avg.), which are obtained as follows: for each test image we average the softmax outputs for the original image, shifted images (9×9 shifts), horizontally flipped image and scaled images (scaling factors are 0.8 and 1.2). Note that this method requires approximately 80 forward passes through a network to obtain a prediction.

Classification accuracies on the test set and on the augmented test set for CNNs trained with invariance regularization and for CNNs trained without the invariance regularization are reported in Table 1. The training set accuracies were 100% or very close to 100% in all cases. First, we observe that invariance regularization leads to a lower GE (a higher accuracy) in all cases. Moreover, testing with the augmented test set is even more robust and leads to a lower GE for both, the regularized and the non-regularized CNNs. Note however, that CNNs trained with explicit invariance regularization (except when 2500 training samples are used) performs better or on par with a non-regularized network evaluated on the augmented test set, where testing with the augmented test set is approximately 80 times more expensive than conventional testing with a single image. This experiment verifies the hypothesis that enforcing the network invariance explicitly can lead to a smaller GE.

The ratio of the GEs of the CNN trained with data augmentation and the invariance regularization and the CNN trained without data augmentation are between 1.5 and 2. Note that the theory from Section 3 does not apply directly as (i) the CNN trained without data augmentation is already (partially) invariant to translations due to its convolutional structure with pooling (Boureau et al., 2010; Bruna et al., 2013); (ii) the CNNs trained with data augmentation and invariance regularization are not perfectly invariant as defined in Definition 4, but only approximately invariant.

Table 1: Classification accuracy [%] on CIFAR-10.

	number of training samples				
	2500	5000	10000	20000	50000
No reg.	68.71	76.74	85.17	87.15	93.65
Inv. Reg.	69.32	79.08	86.69	88.14	94.50
No reg. + avg.	70.59	78.40	86.05	88.13	94.26
Inv. Reg. + avg.	70.71	79.65	86.96	88.98	94.78

6 Discussion and Conclusion

We have formally demonstrated that the GE of an invariant learning algorithm can be much smaller than the GE of a non-invariant learning algorithm, provided that the input space can be factorized into a product of a transformation set and a base space, where the covering number of the base space is much smaller than the covering number of the input space. This work offers an important foundation for the study of the GE of learning algorithms, such as CNNs and their extensions, that leverage symmetries in the data.

Our assumption in this work is that the set of transformations $\mathcal{T}$ is discrete. A more general approach would be to assume that the set of transformations $\mathcal{T}$ is continuous. We conjecture that current results can be extended to such cases by an appropriate covering of $\mathcal{T}$. Second, we have assumed that a learning method is perfectly invariant. The notion can be extended to approximately invariant learning methods and bounds of the same form as in (14) can be derived for this case.

Acknowledgements

The work of J. Sokolić and M. R. D. Rodrigues was supported in part by EPSRC under grant EP/K033166/1. The work of R. Giryes was supported in part by GIF. The work of G. Sapiro was supported in part by NSF, ONR, ARO, and NGA.

References

Y. S. Abu-Mostafa. Hints and the VC dimension. *Neural Computation*, 5(2):278–288, 1993.

N. Alon, S. Ben-David, N. Cesa-Bianchi, and D. Haussler. Scale-sensitive dimensions, uniform convergence, and learnability. *Journal of the ACM*, 44(4): 615–631, July 1997.

F. Anselmi, J. Z. Leibo, L. Rosasco, J. Mutch, A. Tacchetti, and T. Poggio. Unsupervised learning of invariant representations with low sample complexity: the magic of sensory cortex or a new framework for machine learning? *arXiv:1311.4158v5*, 2014.

F. Anselmi, L. Rosasco, and T. Poggio. On invariance and selectivity in representation learning. *Information and Inference*, 5(2):134–158, 2016.

P. L. Bartlett and S. Mendelson. Rademacher and Gaussian complexities: risk bounds and structural results. *Journal of Machine Learning Research (JMLR)*, 3:463–482, 2002.

Y.-L. Boureau, J. Ponce, and Y. LeCun. A theoretical analysis of feature pooling in visual recognition. *Proceedings of the 27th International Conference on Machine Learning (ICML)*, pages 111–118, 2010.

O. Bousquet and A. Elisseeff. Stability and generalization. *Journal of Machine Learning Research*, 2: 499–526, 2002.

J. Bruna and S. Mallat. Invariant scattering convolution networks. *IEEE Transactions on Pattern Analysis and Machine Intellignce*, 35(8):1872–1886, Mar. 2012.

J. Bruna, A. Szlam, and Y. LeCun. Learning stable group invariant representations with convolutional networks. *International Conference on Learning Representations (ICLR)*, 2013.

N. Cohen and A. Shashua. Inductive bias of deep convolutional networks through pooling geometry. *arXiv:1605.06743*, 2016.

T. S. Cohen and M. Welling. Group equivariant convolutional networks. *Proceedings of the 33rd International Conference on Machine Learning (ICML)*, pages 2990–2999, 2016.

C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–279, 1995.

S. Dieleman, J. De Fauw, and K. Kavukcuoglu. Exploiting cyclic symmetry in convolutional neural networks. *arXiv:1602.02660*, 2016.

R. Gens and P. Domingos. Deep symmetry networks. *Advances in Neural Information Processing Systems 27*, pages 2537–2545, 2014.

T. Hofmann, B. Schölkopf, and A. J. Smola. Kernel methods in machine learning. *The Annals of Statistics*, 36(3):1171–1220, 2008.

J. Huang, Q. Qiu, G. Sapiro, and R. Calderbank. Discriminative robust transformation learning. *Advances in Neural Information Processing Systems (NIPS)*, pages 1333–1341, 2015.

M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu. Spatial transformer networks. *Advances in Neural Information Processing Systems 29*, pages 2017–2025, 2015.

Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, May 2015.

K. Lenc and A. Vedaldi. Understanding image representations by measuring their equivariance and equivalence. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 991–999, 2015.

B. Neyshabur, R. Tomioka, and N. Srebro. Norm-based capacity control in neural networks. *Proceedings of The 28th Conference on Learning Theory (COLT)*, pages 1376–1401, 2015.

T. Poggio, J. Mutch, J. Leibo, L. Rosasco, and A. Tacchetti. The computational magic of the ventral stream: sketch of a theory (and why some deep architectures work). *CSAIL Tecnical Reports (MIT-CSAIL-TR-2012-035)*, Dec. 2012.

P. Shah and V. Chandrasekaran. Group symmetry and covariance regularization. *Electronic Journal of Statistics*, 6:1600–1640, 2012.

S. Shalev-Shwartz and S. Ben-David. *Understanding machine learning: from theory to algorithms*. Cambridge University Press, 2014.

L. Sifre and S. Mallat. Rotation, scaling and deformation invariant scattering for texture discrimination. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1233–1240, 2013.

P. Y. Simard, D. Steinkraus, and J. C. Platt. Best practices for convolutional neural networks applied to visual document analysis. *International Conference on Document Analysis and Recognition (ICDAR)*, pages 958–962, 2003.

S. Soatto and A. Chiuso. Visual representations: defining properties and deep approximations. *International Conference on Learning Representations (ICLR)*, 2016.

J. Sokolić, R. Giryes, G. Sapiro, and M. R. D. Rodrigues. Robust large margin deep neural networks. *arXiv:1605.08254*, 2016.

I. Soloveychik, D. Trushin, and A. Wiesel. Group symmetric robust covariance estimation. *IEEE Trans-*

actions on Signal Processing, 64(1):244–257, Jan. 2016.

V. N. Vapnik. An overview of statistical learning theory. *IEEE Transactions on Neural Networks*, 10(5): 988–999, Sept. 1999.

V. N. Vapnik and A. J. Chervonenkis. The necessary and sufficient conditions for consistency of the method of empirical risk. *Pattern Recognition and Image Analysis*, 1(3):284–305, 1991.

T. Wiatowski and H. Bölcskei. A mathematical theory of deep convolutional neural networks for feature extraction. *arXiv:1512.06293*, 2015.

H. Xu and S. Mannor. Robustness and generalization. *Machine Learning*, 86(3):391–423, 2012.

S. Zagoruyko and N. Komodakis. Wide residual networks. *arXiv:1605.07146*, 2016.