# Direct Modeling of Complex Invariances for Visual Object Features

**Ka-yu Hui** KAYUHUI@GMAIL.COM
Department of Information Engineering, The Chinese University of Hong Kong, Shatin, Hong Kong

## Abstract

View-invariant object representations created from feature pooling networks have been widely adopted in state-of-the-art visual recognition systems. Recently, the research community seeks to improve these view-invariant representations further by additional invariance and receptive field learning, or by taking on the challenge of processing massive amounts of learning data. In this paper we consider an alternate strategy of directly modeling complex invariances of object features. While this may sound like a naive and inferior approach, our experiments show that this approach can achieve competitive and state-of-the-art accuracy on visual recognition data sets such as CIFAR-10 and STL-10. We present an highly applicable dictionary learning algorithm on complex invariances that can be used in most feature pooling network settings. It also has the merits of simplicity and requires no additional tuning. We also discuss the implication of our experiment results concerning recent observations on the usefulness of pre-trained features, and the role of direct invariance modeling in invariance learning.

## 1. Introduction

The learning of view invariant representation has been a widely adopted strategy in recent state-of-the-art visual recognition systems. The value of such representation rests on their ability to distinguish different objects of concern despite large amounts of different viewing angles, deformations, lighting conditions and other view related complications in object images.

Learning these representations is not trivial, and the most successful approach is to employ a feature pooling network, which comes primarily in two archetypes. The first type of these feature pooling networks is multi-layered and convolutional, the most representative being convolutional neural network (LeCun et al., 1998; 2004) and its related deep variants (Lee et al., 2008; 2009). A host of other supervised feature pooling networks (Serre et al., 2005; Pinto et al., 2008), mostly inspired by the structure of the human visual cortex, carry a similar architecture. The second type of these feature pooling networks is more flat in structure, and are primarily descendants of the simple bag-of-features methods (Csurka et al., 2004). They commonly use spatial pyramids of some sort (Lazebnik et al., 2006; Yang et al., 2009; Yu & Zhang, 2010), and operate on image patches.

Despite their differences in implementation, these feature pooling networks share at least one key element - the use of spatial pooling, which provides local translational invariance for object features. While their empirical performance is impressive, the view invariance provided by local translational invariance remains limited. Some recent work in the research community have thus been exploring ways to go beyond local translation invariance to improve representation learning in this regard.

There are at least three major approaches under exploration. The first is invariance learning, which involves the learning of complex invariances, i.e. invariances other than translational invariance, from extra data such as video(Zou et al., 2012; Gregor & LeCun, 2011) or by exploiting innate statistics of object images (Coates et al., 2012; Le et al., 2010; 2011). The second approach is massive parallelism (Ciresan et al., 2010; Krizhevsky et al., 2012; Le et al., 2012), which involves scaling up the network and attempts to push the limits of these feature pooling networks using large amount of data and parallelism. The third approach is receptive field learning (Jia et al., 2012; Feng et al., 2011), which either breaks from the tried-and-tested rule of locality, or goes beyond just being spatially local (Boureau et al., 2011).

*Proceedings of the 30$^{th}$ International Conference on Machine Learning*, Atlanta, Georgia, USA, 2013. JMLR: W&CP volume 28. Copyright 2013 by the author(s).

In this study we explore, to the best of our knowledge, a rather neglected alternative: the possibility of the direct modeling of complex invariances in object features. We know the head of cat can tilt, and one side of a car looks the same as the other. These knowledge appear trivial but to our feature pooling networks, they are not trivial and are not easily provided by translation invariance alone[1]. We wonder, how would such an approach compare to the three major approaches mentioned above?

Direct modelling of complex view invariances might sound like a naive approach, but we have to point out much success of existing feature pooling networks is attributed to the direct modeling of translational invariance. Our heavy reliance on local translational invariance can be traced to the work of (Hubel & Wiesel, 1962), which discovers the local translation invariance of simple features in the lowest level of the visual cortex. The direct modeling of this particular prior knowledge has turned out to be an effective strategy.

There are other good reasons to consider this alternative. Learning is often costly. Learning implies risk of overtraining, complexity and requirement of data. Indeed, the current direction tends to require considerable additional data, such as video datasets (Zou et al., 2012), or in some cases massive amount of unlabeled data (Le et al., 2012; Coates et al., 2012). Direct modeling clearly has its own merits.

So in this paper we present a novel method of using prior knowledge to create feature dictionaries with complex invariances. We also present an improvement that gives the algorithm a higher applicability. Despite the simplicity of the approach, we are able to obtain state-of-the-art accuracy on standard benchmarks such as CIFAR-10 and STL-10, demonstrating that direct modeling of complex invariances is an equally viable alternative to other recent approaches. The method has the merits of not being susceptible to overtraining, does not complicate parameter tuning, works without additional data, and is easily applicable to different feature pooling networks.

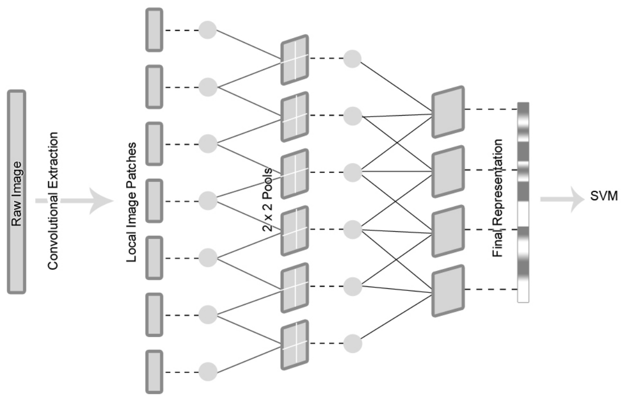

*Figure 1. A simplified depiction of the base system. Dotted lines denote encoding while solid lines denote pooling. Notice that in the actual implementation the network mapping is in 2D and it has considerably higher number of connections.*

## 2. The Base System

We first describe the base system we use in the study. Our base system starts from raw images and the pipeline is shown in Figure 1. It consists of two layers with successively larger receptive field[2] size. Exact size would depend on the task involved but for a 32px $\times$ 32px input image the first layer will have receptive size of 4px and the second 8px.

The overall classification pipeline is very similar to a classical feature pooling network, except we have combined the designs of several notable studies and state-of-the-art designs in the area. This gives us a reasonably good baseline to represent performance of a well-optimized translation-invariance only system. It also serves as an exemplar setup to which our soon proposed algorithm could best be applied.

Each layer involves a coding operation and a pooling operation. Coates el al. (Coates & Ng, 2011a) has recently demonstrated that relatively simple dictionary learning and encoding can already give us sufficiently good performance, and we have adopted a similar implementation. In each layer, we extract local patches convolutionally from the input and encode them using an overcomplete patch dictionary $D$ of size $M$.

[1]While basic feature detectors like SIFT or HOG is by itself rotation and scale invariant, notice that this property does not transfer to the object features constructed. Also, object features are likely to be built upon certain mid level features, so often they are no longer amenable to typical image transforms. This makes complex invariances for object features far more complicated than that of lower level ones.

[2]Receptive field denotes the local area within which coding or pooling operations take place.

The patch dictionary is learned in prior unsupervised from random local patches via the K-means approach and encoded using either triangle activation or sparse coding[3]. The training of the patch dictionary starts from the lowest layer, and then we train the dictionary in the next layer based on results in the previous. This greedy layer-wise strategy is pioneered and demonstrated to be effective by (Hinton et al., 2006).

The encoded representations are highly redundant and are pooled (max or average, no significant difference empirically in our case) across larger receptive fields to form a pooled representation. We will adopt a standard $2 \times 2$ pooling grid which has been shown to be sufficient good in several related studies. Like most feature pooling networks, For instance, the 8px by 8px patches of the first layer are encoded and pooled in a 2x2 grid across a 16px by 16px receptive field[4]. The length of the pooled representation depends on the number of features in the patch dictionary. In our case, it will be four times the number of maps due to the 2x2 grid used.

The encoding and pooling are repeated for each layer, so all layers other than the first will be operating on input from a space dependent on the previous layer's patch dictionary. It is considered a good practice to normalize and whiten the output of each layer. At the end of the pipeline, a simple linear classifier, a linear SVM in this case, will use the output representation to predict the class labels. As a common practice, sometimes the output representation will be appended with the output from the previous layers to maximize its effect to classification accuracy.

### 2.1. Performance of Base System

We shall see that the base system gives competitive performance in many image classification tasks. This gives us confidence that further performance gains will be a result of our proposed improvements, and that they are of interest to state-of-the-art.

As we proceed, we assume the readers are familiar with most of the mentioned and implied principles that drive the performance of the base system. For a review and systematic introduction, please refer to (Coates & Ng, 2011a; Bengio et al., 2012).

*Table 1.* List of complex invariances on object features in natural environments.

| Can be Modeled Directly |
|---|
| Rotation (20 degree), Rotation (45 degree), Rotation (90 degree), Scaling (85%), Vertical Mirror, Horizontal Mirror, Shearing(Left-Right), Shearing(Top-Bottom), Stretching(Left-Right), Stretching(Top-Bottom), Contrast Inversion |
| **Cannot be Modeled Directly** |
| Out-of-Plane Rotations, Warping, Perspective Transforms, Lighting Conditions |

## 3. Algorithm Details

The strategy of direct modeling of complex variance would, first, require prior knowledge on complex variance of object features. A reasonable list of complex invariances is given in Table 1. By reasonable we mean it has to be an intuitive capability of a typical human observer. For instance, a human observer would likely recognize the right side of a car given a sample of its left side. So invariance to horizontal mirroring would be a reasonable candidate. Among the list, some of them are obviously difficult to model directly, like out-of-plane rotation. The rest would constitute our list of candidates to model as invariances and would be subject to elimination by simple experiments.

Formally, let us denote a complex invariance simply as $\mathcal{I}$. We model $\mathcal{I}$ as a set of image transforms $\mathcal{I} = [\mathcal{T}_1 ... \mathcal{T}_K]$ that operates on image patches of size appropriate to the layer. For instance, invariance to slight in-plane rotation could be represented by the two transforms that rotate an image patch clockwise and counter-clockwise by say 20 degrees respectively.

Suppose we are interested in constructing an $\mathcal{I}$-invariant dictionary. Given a patch dictionary $D = [d_1 d_2 ... d_M]$ that corresponds to $M$ activation values, for each feature $d_j \in D$, we would desire its activation $a_j$ to be invariant to transforms in $\mathcal{I}$. One intuitive way would be to extend the concept of spatial pooling to complex invariances. Spatial pooling, in essence, makes a feature translation-invariant by pooling the feature and their locally translated occurrences together. Applying to our case, we would have to pool activations of $d_j$, $\mathcal{T}_1(d_j)$ ... $\mathcal{T}_K(d_j)$ together (via max-pool or otherwise) to form an $\mathcal{I}$-invariant activation

[3]Both encoding are said to perform well under sufficient amount of labeled data. For exact implementation details, see (Coates & Ng, 2011a)

[4]px = pixel

$a'_j$, i.e.

$$a'_j = \max(a_j, a_j^{\mathcal{T}_1}, a_j^{\mathcal{T}_2} ... a_j^{\mathcal{T}_K}); \tag{1}$$

where $a_j^{\mathcal{T}_k}$ would be activation to $\mathcal{T}_k(d_j)$. This response behavior will fit with the known response behavior or neurons in higher layers of visual cortex (Pinto et al., 2008).

This would appear trivial if we are operating on raw pixels - we will just calculate $\mathcal{T}_1(d_j)$ ... $\mathcal{T}_K(d_j)$ and create additional dictionaries. But this can only happen in lowest layer of the feature pooling network. At the lowest layer, however, we are mainly interested in orientation selectivity, not complex invariances. In higher layers where complex invariances would matter, the image patches would be represented in a completely different feature space[5]. To perform direct calculation we would need a set of corresponding transforms $\mathcal{T}'_1$ ... $\mathcal{T}'_K$ that operates in the said feature space, but there are no obvious ways to obtain them.

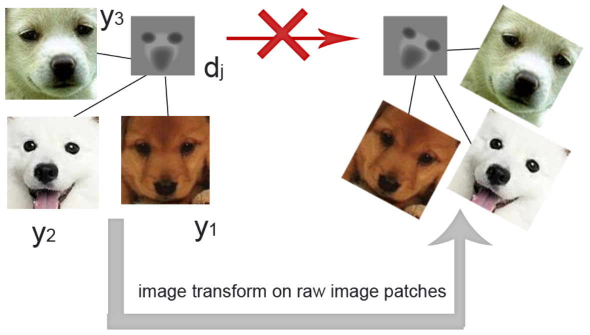

*Figure 2. Conceptual explanation of VIK. In this example, we have a dog-face feature $d_j$ that we wish to apply rotation to, but it is in a high-level representation so there are no obvious means for direct calculation. Instead, during the K-means clustering process in which it is created, we rotate the raw image patches that are members of this particular cluster, and equivalently feature, and utilize them to compute the supposedly rotated cluster in high level representation. We now have a pair of features that we can establish rotation invariance with.*

## 3.1. The View-Invariant K-means (VIK)

This is where our algorithm would enter the scene. This algorithm, which we will call view-invariant K-means (VIK), is modified from the simple K-means dictionary learning in (Coates & Ng, 2011a). Assume that we are given a set of image patches $y_1 ... y_n$ extracted from the receptive fields of a layer of concern. The image patches are expected to have gone through one or more lower layers, so each $y_i$ is already converted to a pooled, overcomplete representation $f(y_i)$ where $f$ is a mapping that provides an abstraction over the process. Each representation in $Y = [f(y_1) ... f(y_n)]$ is assumed normalized with zero mean and whitened, as mentioned in our description of the base system. The learning algorithm is as follows:

1. Learn a patch dictionary $D$ of desired size $M < n$ from
$$\min_{D,C} ||DC - Y||^2 \tag{2}$$
such that $||d_j|| = 1$ and $||C_i||_0 = 1$ for all $i$,$j$.

   Here $||$ denotes l2 norm and $||_0$ denotes zero "norm" that simply counts the number of non-zeroes. The optimization can be solved by alternating between the dictionary $D$ and the membership matrix $C$ as in regular K-means clustering.

2. For each image transform $\mathcal{T}_k$ in designated complex invariance $\mathcal{I}$, compute transformed input matrix $Y^{\mathcal{T}_k} = [f(\mathcal{T}_k(y_1)) ... f(\mathcal{T}_k(y_n))]$, which is how the input patches would be represented if they are all under transform $\mathcal{T}_k$.

3. Produce view dictionary $D^{\mathcal{T}_k}$ of size $M$ by using the transpose of the membership function:
$$D^{\mathcal{T}_k} = Y^{\mathcal{T}_k} C^T / \text{diag}(\mathbf{1}^T C^T) \tag{3}$$
Here each feature $d_j^{\mathcal{T}_k}$ is normalized and would be a transformed version of $d_j$, i.e. $\approx \mathcal{T}'_k(d_j)$.

The result of the algorithm is $K$ view dictionaries $D^{\mathcal{T}_1} ... D^{\mathcal{T}_K}$ for the $K$ image transforms in $\mathcal{I}$. They are, together with the original $D$, called a $\mathcal{I}$-invariant version of dictionary $D$. A conceptual visualization of the algorithm is shown in Figure 2.

To encode an input using the $\mathcal{I}$-invariant dictionary $[D\ D^{\mathcal{T}_1} ... D^{\mathcal{T}_K}]$ we compute activation $a_j$ for $j = 1 ... M$ by an "in-dictionary" pooling:

$$\max(a_j, a_j^{\mathcal{T}_1}, a_j^{\mathcal{T}_2} ... a_j^{\mathcal{T}_K}); \tag{4}$$

where $a_j^{\mathcal{T}_1}$ is the activation of $d_j^{\mathcal{T}_k}$. This echoes our intuition in (1). It works for both triangle coding or sparse coding, but for sparse coding sum pooling works much better.

By inspection, this "trick" on K-means should work well for the linear image transforms we have shortlisted. To confirm the effectiveness of the method, let us proceed to conduct a preliminary experiment on the CIFAR10 data set.

[5]More accurately, it will be represented in a feature space defined by previous layer's dictionary and pooling function.

*Table 2.* Preliminary experiments on CIFAR-10, 400 examples per class.

| Method | Dict Size | Accuracy |
|---|---|---|
| Multi-VIK | 6400 | 70.5% ± 0.8% |
| Base system | 1600 | 68.7% ± 0.3% |
| Base system (M=6000) | 6000 | 70.3% ± 0.4% |

## 3.2. Preliminary Experiments on CIFAR-10

The CIFAR-10 dataset[6] contains 50,000 32px × 32px images in 10 categories as training data and 10,000 testing data. The first layer was setup with 4px × 4px receptive fields and dictionary of size 200, and the second layer 8px × 8px receptive fields and dictionary of size 400. The rests followed what we have described in the base system. For our purpose, we limited ourselves to 400 labeled example per class to reduce the influence of the SVM classifier on the final accuracy relative to that of the quality of the features.

In the base system, the object features are mainly in the output of the second layer, which is the end of the pipeline. To accommodate our proposed view-invariant K-means (VIK) learning, we constructed a new layer on top. We introduced a third layer with receptive field size of 16px × 16px, a reasonable size for an object feature, e.g. a cat's head[7] The first two layers would remain to be encoded using triangle activation based on a patch dictionary learned via K-means.

For the new third layer we learn a patch dictionary $D$ as usual. It was of size 1600, sufficiently large for our purpose. We then went through the list in Table 1 one by one. For each complex invariance $\mathcal{I}$ we constructed an $\mathcal{I}$-invariant version of $D$ denoted by $D_{\mathcal{I}}$ and we used the combined output of $D$ and $D_{\mathcal{I}}$ as final representation. We then measured the accuracy of the setup on CIFAR testing set by using several random folds of training set (400 example per class).

We identified four complex invariances that give the best gains in accuracy: rotation (45 degrees), rotation (90 degrees), horizontal flip, and contrast inversion. Some others gave little to no gains, such as vertical flip and shearing. On hindsight this result is rather intuitive. You rarely see, say, a dog's head going upside down, or part of a car sheared, so invariance to these transforms would likely be less useful. Combining the four best $D_{\mathcal{I}}$ into what we call a **multi-VIK dictionary** $D'$ gave us a significant gain on test accuracy, as seen in Table 2.

[6]http://www.cs.toronto.edu/ kriz/cifar.html

[7]This assumption can only hold true for datasets like CIFAR-10 and STL-10, where size of objects of concerned are relatively controlled and each image contains only one dominant object. This helps significantly in experimenting with view invariance, like limiting complex invariances to one specific layer here. Without this assumption, as in the case of PASCAL and Caltech-101, the pooling network has to be much more complicated. For a more detailed reasoning on choice of datasets for study of invariance, see (Pinto et al., 2008).

*Table 3.* Classification accuracies for CIFAR-10, 400 examples per class.

| Method | Accuracy |
|---|---|
| Mix-VIK | **72.6**% ± 0.7% |
| Base System | 70.3% ± 0.4% |
| (Coates & Ng, 2011b), 3 layers | 70.7% ± 0.7% |
| (Coates & Ng, 2011a), Sparse | 66.4% ± 0.8% |
| (Coates & Ng, 2011a), VQ | 64.4% ± 1.0% |

The result of this simple experiment is surprising. Our modeling of the short-listed complex invariances would appear crude. We have not optimized any of their parameters. However, the preliminary results has shown that the simple model can already give meaningful results.

## 3.3. Mix-VIK: Improving the Multi-VIK Dictionary

Now we have shown that direct modeling of complex variance can indeed improve a feature pooling system's accuracy in a visual recognition task. A key weakness to this naive implementation is the multi-VIK dictionary $D'$, as constructed in the above session, consists of highly correlated features. For any original feature $d$ there will be multiple versions of the same feature with different invariance. A dictionary of similar size to $D'$ created using generic K-means would give us, in general, a more diverse basis.

In the case above, $D'$ would alone contribute an output representation size of $1600 \times 4 \times 4 = 25{,}600$ due to the four invariances and $2 \times 2$ pooling. A generic dictionary of size 6400 would give a similarly sized output representation. In Table 2 we can see that an increased dictionary size alone does lead to similar gain in performance. In such a case, the multi-VIK would be an interesting proof of concept, but it would not be too useful in practice.

This motivates us to seek a more efficient approach. We will call this approach mix-VIK dictionary learning. To create a mix-VIK dictionary, we would perform the following steps:

1. Create a large generic K-means patch dictionary $D$. Say, in our example case here, 6400 in size.

2. For each feature $d_j$, randomly select a complex invariance $\mathcal{I}_p$ from the set of complex invariances $\mathcal{I}_1, \mathcal{I}_2 \dots \mathcal{I}_N$ that we have selected to build invariance to. In our example case, we will be randomly selecting from the two rotations, horizontal flip, and contrast inversion.

3. Using the VIK learning algorithm above, replace feature $d_j$ with a $\mathcal{I}_p$-invariant version of $d_j$. Notice that different features will have different in-dictionary pooling as defined in (4).

4. Repeat until a designated fraction or all original features in $D$ are converted.

The mix-VIK dictionary created using this simple strategy will have the benefit of having the same diverse basis of a similarly sized generic dictionary but at the same time enhanced with a variety of complex invariances. Repeating our experiment on CIFAR-10 we achieve an accuracy of 72.6%, significantly surpassing the gain provided by a similarly sized multi-VIK dictionary.

The mix-VIK dictionary is a clear improvement over the multi-VIK dictionary. When the linearly increased computation cost is affordable, using a mix-VIK dictionary has almost no downside compared to using a conventional K-means one.

## 4. Results on Benchmark Datasets

Now we should attempt to fine tune our feature pooling network and the mix-VIK dictionaries to see how its performance compare to other related and state-of-the-art systems. As a common practice, one would append output from lower layers to the final output to give maximal performance, so we followed the practice. Using cross-validation to fine-tune parameters in various level, we report our performance on CIFAR-10 using partial data (400 example per class) in Table 3.

Our accuracy on the dataset using a mix-VIK dictionary of 6400 in size is 72.6%, which is a significant improvement over the best published result under similar setup. This result is particularly notable because the main difference between our system and the other systems in comparison is the mix-VIK dictionary, which requires no additional data to train and has very few free parameters.

*Table 4.* Classification accuracies for STL-10, in comparison with state-of-the-art and competitive models.

| Method | Accuracy |
|---|---|
| Mix-VIK | **63.7**% |
| Base system (M=6000) | 59.6% |
| (Zou et al., 2012) | 61% |
| (Coates & Ng, 2011b) | 60.1% |
| (Coates & Ng, 2011a) | 59% |
| (Le et al., 2011) | 52.9% |

### 4.1. Results on STL-10

To facilitate comparison with other studies, we also utilized the STL-10 dataset (Coates et al., 2010), which is more tailored to evaluating feature pooling systems, particularly those using unsupervised feature learning. The 10 object class dataset contains an unlabeled set of 100,000 object images for unsupervised training purposes, and a much smaller training set to restrict the amount of labeled data used to 100 samples per class for each training fold. We used the same system as in CIFAR-10 and retrained the system on the unlabeled data set provided. To accommodate the even smaller labeled data set, we switched our encoding in the third layer to use sparse coding to reduce overtraining.

Our results are reported in Table 4. We have the very encouraging result of achieving state-of-the-art performance in the STL-10 dataset. A key comparison is the best result so far among other similar research, reported by (Zou et al., 2012), which involved learning of complex invariances from temporal information in additional video data using a simulated fixation strategy. We achieved similar improvements over our own baseline (4.1%). The result is clearly surprising - while we have utilized a larger dictionary size, we have used no additional data like video or any kind. We have learned the mix-VIK dictionary based only on crude prior knowledge like rotation transforms and mirroring, and the learning algorithm itself is easily implementable. This suggests that direct modeling of complex invariance in object features is a competitive and viable strategy to invariance learning. We will follow up on this surprising result in the discussion.

### 4.2. Results on Full Training Set, CIFAR-10

Here we wish to point out that there is a curious lack of explicit mention of one intuition in our representation learning literature: that view invariance is only particularly useful when amount of labeled training data is

*Table 5.* Classification accuracies for CIFAR-10, full dataset, in comparison with state-of-the-art (without data augmentation) and competitive models.

| Method | Accuracy |
|---|---|
| Our Method | **81.9%** |
| Base system (M=6000) | 81.6% |
| (Jia et al., 2012) | 83.11% |
| (Coates & Ng, 2011b) | 82% |
| (Krizhevsky, 2010) | 78.9% |
| (Yu & Zhang, 2010) | 74.5% |

small. Consider the imaginary case in which we have asymptotically large amount of labeled data such that every view of the objects of concern is available as an example. In such a case view invariance can be considered unnecessary. So conversely if we have only very few examples, we would have a much stronger desire to relate a limited amount of data to potentially many other unseen views of the same object. In such a case, highly view invariant features would be important.

In this study our focus is complex view invariance, our priority is thus to experiment with limited amount of labeled training data, as we just did. So if we now apply our system and train it using a full CIFAR-10 training set, which contains 5,000 examples per category, we should expect the benefit of the mix-VIK dictionary to decline. Furthermore, invariance implies a reduction in selectivity. So when the view invariance of the object features are becoming less useful in large amount of labeled data, will it become a liability?

Our experiment using full training set of CIFAR-10 is reported in Table 5. As we can see, our strategy remained highly competitive to state-of-the-art, only with the improvement over the base system being much lower. This confirms our prediction. Fortunately, the mix-VIK dictionary did not appear to have negative impact to the performance.

We believe this finding has deeper implications than it appears. We will follow up in discussion in Section 5.2.

On a sidenote, we actually sought to verify that our performance gain is indeed due to the modeling of complex invariances. We tried our system on the MINST dataset of digits, and none of our mix-VIK dictionaries brought improvement to our baseline performance. This is a positive result, because for digits, complex invariances should not apply at all(e.g. we do not write a 3 rotated by 90 degress), so the (lack of) results fit with our expectation.

## 5. Discussion

As far as we know, this is the first in-depth exploration and empirical report on the effectiveness of direct modeling of complex invariances in object features. The algorithm proposed and its improved version are simple to implement, and requires no additional data or parameter tuning. It only requires very simple prior knowledge so that we can have a set of image transforms as candidate invariances in the beginning. While this is only an early exploration, the results are definitely encouraging and the direct modeling of complex invariance should warrant further exploration.

We believe the study has other implications that are related to a number of works in the area of representation learning, particularly studies in the three main approaches we have mentioned in the introduction. We shall go through each approach one by one.

### 5.1. To Learn or Not to Learn?

In Section 4.1 we see that our direct modeling of complex invariance has produced similar gains in accuracy compared to learned invariances in (Zou et al., 2012). It is, however, unjustified to say that direct modeling of view invariance is a substitute of invariance learning because both approaches are in early exploration stage. What we can learn from the result, however, is that direct modeling is more effective than one's intuition would suggest, and that we wonder where the line is when we seek to learn invariance from data.

Indeed, in the study we specifically compared to (Zou et al., 2012), the fixation tracking is meant to improve learning by, in principle, not wasting effort to learn translation invariance so more complex invariances can emerge. Would the invariance learning benefit from focusing on learning what is beyond the crude complex invariances we have directly modeled as well? We believe that would be an interesting direction to look into.

The other possibility is that a mix-VIK dictionary should intuitively contain redundant, or simply useless view-invariant features. It would be desirable to eliminate useless features in the VIK dictionary so we can maximize the effective of our final representation. Could the established findings from invariance learning studies that exploit innate data characteristic, such as multi-way locality (Boureau et al., 2011) or its innate sparsity (Le et al., 2010; 2011; 2012), help in selecting the best complex invariant features?

Incorporating prior knowledge has been known to be a rather effective way to improve machine learning systems in general. Through our empirical results, we can claim that using our prior knowledge on complex invariance is more reliable than we would intuitively believe, and could well be a tool that algorithm designers might find useful.

### 5.2. Unsupervised Learning not Useful?

There are recent observations (Krizhevsky et al., 2012; Ciresan et al., 2010; Bengio et al., 2012) that unsupervised representation learning does not seem to bring improvement to a visual recognition task when significant amount of labeled data is available or made available by data augmentation. In our experiment with full training set, CIFAR-10, we see a decline of the benefits of view invariance in features as amount of labeled data goes up. Since a primary goal of most unsupervised representation learning is view invariance of features, our experimental results could be related to this phenomenon.

Of course, the long pipeline of most feature pooling networks does make it hard to isolate benefits of different elements. However, in the above studies, the amount of labeled data is multiples of original training data, or far more. It is reasonable to speculate that with such amount of labeled data, the effect of view invariance would be almost negligible.

This insight would be more productive if we realize that visual tasks with few training samples should be considered a different case compared with those with many training samples, each with their own applications. Humans, on the other hand, are masters of view invariance and low-sample learning. One could not imagine if a human would, say, need thousands of labeled samples to recognize their own possessions, or new born babies.

We would encourage fellow researchers in the field to experiment on both full training set and partial training set (400 examples per class with at least 10 random folds to eliminate noises) when they experiment on CIFAR-10 dataset. The dataset offers an opportunity to see difference and tradeoff between the two learning cases that is not possible in, say, STL-10 or some other datasets.

### 5.3. Applicability of VIK Learning

One of the strengths of VIK learning is that a mix-VIK dictionary works and behaves like any typical dictionary in a feature pooling network except it provides usable complex invariance at some additional linear cost. This gives the approach the merit of being applicable to any other designs where a conventional patch dictionary is used. For example, recent works in receptive field learning we have mentioned in the introduction, e.g. (Jia et al., 2012), should be eligible to mix-VIK learning to get further gain in accuracy because receptive field learning is, intuitively, largely independent of dictionary learning.

Using full training set, CIFAR-10, we test our system on the task by adding a extra $3 \times 3$ pooling layer 3, which can be considered a receptive field improvement. We are successful in getting around 2% accuracy improvement with some fine tuning on a limited mix-VIK dictionary[8], supporting that gains from receptive field improvements are indeed independent of gains in complex view invariance.

The costs of independently performing receptive field learning and complex invariance modeling may appear formidable. A rigorous analysis or a proposal to lessen the cost, however, is beyond the scope of this paper. A key insight is that while it may sound otherwise, direct modeling of complex invariance is not a substitute of other approaches in representation learning. We believe, via our design, it is sufficiently self-contained to be a handy tool in many cases.

## 6. Conclusion

In this paper, we explored the possibility of directly modeling complex invariances in object features and employ it directly in visual recognition tasks. While this strategy may initially appear to be a naive approach compared to invariance learning, we discover that it can bring surprisingly good results. By extending the concept of spatial pooling to complex invariance, we proposed a novel dictionary learning method called view-invariant K-means (VIK). We also proposed an improved algorithm that allows us to create what we call a mix-VIK dictionary, with which we achieve state-of-arts results in STL-10 dataset, best published results in CIFAR-10 with partial data, and highly competitive results with full data. The success of the experiments suggests that direct modeling of complex invariance is more reliable than we would intuitively believe, and we should rethink its role in representation learning. We have also related our experimental results to a recent observation in large-scale supervised deep networks, and suggested several possible future works for invariance learning.

[8]The denser receptive field is a heavy burden on the representation size and hence a heavy cost to our subsequent supervised training process. For this reason we were not able use a complete VIK-dictionary.

## References

Bengio, Yoshua, Courville, Aaron C., and Vincent, Pascal. Unsupervised feature learning and deep learning: A review and new perspectives. *CoRR*, abs/1206.5538, 2012.

Boureau, Y., Le Roux, N., Bach, F., Ponce, J., and LeCun, Y. Ask the locals: multi-way local pooling for image recognition. In *ICCV*, 2011.

Ciresan, D.C., Meier, U., Gambardella, L.M., and Schmidhuber, J. Deep, big, simple neural nets for handwritten digit recognition. *Neural computation*, 22, 2010.

Coates, A. and Ng, A.Y. The importance of encoding versus training with sparse coding and vector quantization. In *ICML*, 2011a.

Coates, A. and Ng, A.Y. Selecting receptive fields in deep networks. *Advances in Neural Information Processing Systems*, 24, 2011b.

Coates, A., Lee, H., and Ng, A.Y. An analysis of single-layer networks in unsupervised feature learning. In *AISTATS*, 2010.

Coates, A., Karpathy, A., and Ng, A. Emergence of object-selective features in unsupervised feature learning. In *NIPS*, 2012.

Csurka, G., Dance, C., Fan, L., Willamowski, J., and Bray, C. Visual categorization with bags of keypoints. In *Workshop on Statistical Learning in Computer Vision, ECCV*, 2004.

Feng, J., Ni, B., Tian, Q., and Yan, S. Geometric l p-norm feature pooling for image classification. In *CVPR*, 2011.

Gregor, K. and LeCun, Y. Efficient learning of sparse invariant representations. *arXiv preprint arXiv:1105.5307*, 2011.

Hinton, G.E., Osindero, S., and Teh, Y.W. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7), 2006.

Hubel, D. H. and Wiesel, T. N. Receptive fields, binocular interaction and functional architecture in the cats visual cortex. *J Physiol*, 160:106154, 1962.

Jia, Y., Huang, C., and Darrell, T. Beyond spatial pyramids: Receptive field learning for pooled image features. In *CVPR*, 2012.

Krizhevsky, A. Convolutional deep belief networks on cifar-10. *Unpublished manuscript*, 2010.

Krizhevsky, A., Sutskever, I., and Hinton, G. Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*, 25, 2012.

Lazebnik, S., Schmid, C., and Ponce, J. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*, 2006.

Le, Q., Ranzato, M.A., Monga, R., Devin, M., Chen, K., Corrado, G., Dean, J., and Ng, A.Y. Building high-level features using large scale unsupervised learning. In *ICML*, 2012.

Le, Q.V., Ngiam, J., Chen, Z., Chia, D., Koh, P.W., and Ng, A.Y. Tiled convolutional neural networks. In *NIPS*, 2010.

Le, Q.V., Karpenko, A., Ngiam, J., and Ng, A.Y. Ica with reconstruction cost for efficient overcomplete feature learning. In *NIPS*, 2011.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86, 1998.

LeCun, Y., Huang, F.J., and Bottou, L. Learning methods for generic object recognition with invariance to pose and lighting. In *CVPR*, 2004.

Lee, H., Ekanadham, C., and Ng, A.Y. Sparse deep belief net model for visual area v2. In *NIPS*, 2008.

Lee, H., Grosse, R., Ranganath, R., and Ng, A.Y. Convolutional deep belief networks for scalable unsupervised learning of heirarchical representations. In *ICML*, 2009.

Pinto, N., Cox, D., and Dicarlo, J. Why is real-world visual object recognition hard. *PLoS Computational Biology*, 4(1):151–156, 2008.

Serre, T., Wolf, L., and Poggio, T. Object recognition with features inspired by visual cortex. In *CVPR*, 2005.

Yang, J., Yu, K., Gong, Y., , and Huang, T. Linear spatial pyramid matching using sparse coding for image classification. In *CVPR*, 2009.

Yu, K. and Zhang, T. Improved local coordinate coding using local tangents. In *ICML*, 2010.

Zou, W., Ng, A., Zhu, S., and Yu, K. Deep learning of invariant features via simulated fixations in video. In *NIPS*, 2012.