ARL-TR-8667 • MAR 2019

Using Convolutional Neural Networks to Extract Shift-Invariant Features from Unlabeled Data

by Samuel Edwards and Michael S Lee

NOTICES

Disclaimers

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

Citation of manufacturer's or trade names does not constitute an official endorsement or approval of the use thereof.

Destroy this report when no longer needed. Do not return to the originator.

ARL-TR-8667 ● MAR 2019

Using Convolutional Neural Networks to Extract Shift-Invariant Features from Unlabeled Data

by Samuel Edwards
Parsons, Inc., Columbia, MD

Michael S Lee
Computational and Information Sciences Directorate, CCDC Army Research Laboratory

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED (From - To)
March 2019	Technical Report	June–December 2018

4. TITLE AND SUBTITLE
Using Convolutional Neural Networks to Extract Shift-Invariant Features from Unlabeled Data

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Samuel Edwards and Michael S Lee

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
US Army Combat Capabilities Development Command
Army Research Laboratory*
ATTN: FCDD-RLC-HC
Aberdeen Proving Ground, MD 21005

8. PERFORMING ORGANIZATION REPORT NUMBER
ARL-TR-8667

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES
*The work outlined in this report was performed while the US Army Research Laboratory (ARL) was part of the US Army Research, Development, and Engineering Command (RDECOM). As of 31 January 2019, the organization is now part of the US Army Combat Capabilities Development Command (formerly RDECOM) and is now called CCDC Army Research Laboratory.

14. ABSTRACT
Unsupervised learning on limited data is a challenging task. In this work, we show that shallow what-where autoencoders, first developed as a pretraining tool for supervised classifiers, can also be used for shift-invariant feature extraction. Furthermore, feature vectors (i.e., the bottleneck layer activations), can be clustered to achieve unsupervised segmentation. In order to remove edge artifacts in the segmentation, overcoding is introduced, whereby the decoder only needs to reproduce a cropped version of the encoded signal.

15. SUBJECT TERMS
unsupervised learning, shift-invariant feature extraction, unlabeled data, convolutional what-where autoencoders, dictionary learning

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Samuel Edwards
Unclassified	Unclassified	Unclassified	UU	35	19b. TELEPHONE NUMBER (Include area code)

Contents

List of Figures

List of Tables

Acknowledgments

We would like to thank the US Army Combat Capabilities Development Command's Army Research Laboratory DOD Supercomputing Resource Center for providing computing hours for this work.

1. Introduction

While it can be difficult to draw meaningful conclusions from sparse unlabeled data, extracting informative features carries the potential to glean understanding or recognize patterns that may otherwise go undetected. Many conventional machine learning methods require a substantial amount of data to produce reliably accurate results. However, it may not always be feasible to obtain plentiful data samples. A recent surge of interest in one-shot learning—that is, learning on one or a few samples—has arisen in response to this problem.[1–3] The majority of these techniques involve supervised or semi-supervised techniques. There are many cases, however, in which it may not be practical to confer any labels on the data. Therefore, we desire an unsupervised feature detection algorithm that is able to operate on very few samples of unlabeled data. Superficially, dictionary learning satisfies these stipulations. The dictionary learning algorithm has demonstrated significant utility with the tasks of image classification,[4] facial recognition,[5–7] and unsupervised clustering.[8]

Suppose, however, that the data contains multiple, frequently repeating patterns. Because conventional dictionary learning algorithms are not immune to translations, the resultant feature dictionary contains many nearly redundant patterns that are mixed phases of the same underlying feature. In the similar field of subsequence time series clustering, the mixed phase problem can be so extreme as to yield simple uninformative periodic waveforms as features.[9] Thus, there is a need for invariance to small shifts and distortions in the dictionary.[10–14] A popular approach is modifying the dictionary update step of the K-SVD algorithm[15] to handle all phase shifts of the learned atoms.[11,12,14] The subspace clustering algorithm[16] can be optimized to accommodate training the dictionary on multiple shifted versions of the original signal.[13] It is possible to reduce the computational complexity by using multiple layers of dictionary atoms to build a hierarchical graph structure of atoms with certain shift-invariance constraints.[14] Ranzato et al.[10] developed a method for unsupervised shift-invariant feature extraction that comprises convolutional filters followed by a layer of max-pooling. Originally, this method was only envisioned as a pretraining optimization of features for a supervised classifier. In this work, we connect this method to shallow what-where convolutional autoencoders,[17] demonstrating that the algorithm can be used for shift-invariant dictionary learning in both one and two dimensions. Furthermore, this method doubles as an effective unsupervised classifier on limited data. Various deep architectures of autoencoders have been employed in image segmentation and classification texts;[18–21] our method differentiates itself by including a single convolutional layer that employs what-where switches.[17,22] While a stacked

convolutional what-where autoencoder (WWAE) with small pooling size was shown to be less accurate in unsupervised classification than those without the what-where switches,[22] we show that a shallow convolutional network with a larger max pooling size results in a versatile unsupervised classifier.

The feature extraction methods discussed thus far capture features that are necessary for the reconstruction of the image after some sort of dimensionality reduction, denoted here as *primitive features*. While descriptive, there are many cases in which primitive features can fail to fully encapsulate the flavor of the data. Therefore, in this work, we use a method that captures frequently occurring groups of primitive features that we dub *collective features*. This is readily accomplished by clustering the intermediate activation maps of the encoding convolutional layer in the WWAE. The identification of collective features on the input signal leads to an unsupervised partial segmentation of the signal/image. Typically, the best performing models for semantic segmentation combine image features with object detectors, which can be improved upon by supervised pretraining.[23] With a shallow, completely unsupervised model trained on a single image, we do not achieve the same accuracy, but our method achieves useful segmentation as a side effect of clustering commonly occurring feature groups.

Throughout our analysis of the activations in the intermediate layer, we noticed an edge effect in the convolutional process of the WWAE when breaking the signal into fragments. As the filter slides along, it is not always fully contained within the fragment; the zero-padding here creates edges along each patch. We resolve this issue in a novel approach by imposing that the decoder reconstruct a smaller region than what was encoded—a process we refer to as *overcoding*. This process immensely improves the accuracy of our unsupervised segmentation.

2. Theory

2.1 Primitive Feature Extraction

We define primitive features as those necessary for reconstruction of the input signal after some form of dimensionality reduction, primarily obtained through signal processing algorithms. Primitive features are intended to capture interesting parts of the data set, such as textures or repeated patterns, and provide an informative summary of the input signal. We present an overview of the techniques utilized in this report for primitive feature extraction: dictionary learning and shallow convolutional WWAEs.

2.1.1 Conventional Dictionary Learning

Given an input signal x, the objective of dictionary learning is to compute a matrix D such that the original input may be closely approximated as a linear combination $x = Ds$ of a few atoms of the dictionary—we call s a sparse representation of x. The dimension of the dictionary is equal to that of the input vector in this setting. For an input data set $X = [x_1, x_2, \ldots, x_N]$, we seek to compute a dictionary that minimizes the error between each signal and its approximation such that the sparse representation satisfies a constraint. That is, we find D and s subject to

$$\min_{D,s_i} \sum_i \|X_i - Ds_i\|_2^2 + \lambda \|s_i\|_0, \quad (1)$$

where the number of nonzero s_i is ideally much smaller than the length of vector s. It has been shown that applying the ℓ_1-norm to s can provide an identical solution as shown in Eq. 1,[24] this allows the problem to be solved in polynomial time. We also impose that each column in D, often called an *atom*, is a unit vector.

Dictionary learning is typically accomplished by alternating between fixing the dictionary D and sparse representation s while minimizing the other. The best possible sparse representation for a fixed D is calculated through algorithms such as Orthogonal Matching Pursuit (OMP)[25] or Focal Underdetermined System Solver (FOCUSS).[26] The dictionary is then updated based on this new sparse representation. In the Method of Optimal Directions (MOD), the sparse representation is held constant as D is renormalized.[27] The K-SVD algorithm[15] updates each column in D separately and appropriately changes the relevant coefficients in s to reduce the mean squared error, accelerating convergence. Alternatively, dictionary learning can be accomplished by assuming that signals generated from the same dictionary elements should lie in the subspace spanned by those atoms. In subspace clustering,[16] the input signals are clustered based on the subspaces of dictionary atoms that contain them. The dictionary is then composed of the collection of vectors that span each of these subspaces.

2.1.2 Shift-Invariant Dictionary Learning

Shift-invariant dictionary learning is employed to capture local patterns that appear throughout a signal.[11, 28] This prevents frequently appearing features from clogging the dictionary and multi-phase mixtures from smearing the learned features, allowing for a cleaner analysis of the input signal. We allow the dimension of dictionary atoms to be smaller than that of the input signal. Thus, we must account for all possible shifts of each dictionary atom d_j along the input signal x_i; let $t_{i,j}$ represent the location of atom d_j along the signal x_i. Denote $\tau(d_j, t_{i,j})$ to be the

vector with dimension equivalent to x_i that is null everywhere except for a copy of d_j that begins at $t_{i,j}$. Therefore, our new objective function becomes

$$\min_{d_j, t_{i,j}, s_{i,j}} \left\| X - \sum_i \sum_j s_{i,j} \tau(d_j, t_{i,j}) \right\|_2^2 . \quad (2)$$

The K-SVD algorithm may be extended to solve the shift-invariant case;[11,12,14] here, each atom is updated individually depending on the existence of overlap and appropriate changes are made to the corresponding sparse representation. Another algorithm[28] updates s through coordinate descent while fixing D. It then employs coordinate descent and the Lagrange dual method[29] to upgrade D while freezing s. The subspace clustering method may be adapted to train on multiple phase shifts of the input signals with an added subspace pruning phase to handle the vast amount of training data.[13]

2.1.3 Convolutional Autoencoders

A convolutional autoencoder follows an encoder-decoder paradigm—that is, an input signal is rebuilt after first being transformed into a lower-dimensional representation. An input signal X is first encoded into its latent representation r. The latent representation of the i-th feature map is $r_i = \sigma(W_i * X + b_i)$, where W_i and b_i are the respective weight and bias vectors and σ is a (typically nonlinear) activation function. Recall that the convolution of two signals $f(t)$ and $g(t)$ is defined as $f(t) * g(t) = \int_{-\infty}^{\infty} f(t)g(t-u)du$. The signal is then reconstructed (or decoded) through the following

$$Y = \sigma\left(\sum_i W_i' * r_i + c\right), \quad (3)$$

where W_i' is simply W_i flipped over both dimensions. The weights are shared throughout the input, allowing spatial locality to be preserved. The reconstruction is learned through the minimization of a loss function.

A pooling layer is often added after convolution in order to reduce the dimension of the feature maps and induce translation invariance. For example, max-pooling of the latent representations summarizes the maximum value of a rectangular region. Small degrees of translational invariance are imposed by this down-sampling as small shifts in the input would not change the maximum in a certain region. This is beneficial when determining the presence but not exact location of a certain feature. Recent literature has suggested that the max-pooling layer may be entirely replaced by convolutional layers with larger stride with no loss in accuracy

in the supervised case;[30] however, the features captured by these models tend to be tangled and incoherent, as we will demonstrate.

In WWAEs,[17,31,32] pooling layers in the encoder of an autoencoder are accompanied by the appropriate unpooling layer in the decoding phase. Unpooling is typically conducted by restoring the maximum value to the entire region from which it was extracted, diluting exact spatial information. To remedy this, the location (the "switch" or "where") of the maximum value obtained from max-pooling (the "what") can be passed to the decoder, restoring the maximum to its original position in the input signal. A diagram of this network is depicted in Fig. 1. These WWAEs have been shown to produce better reconstruction quality than those of simple upsampling. This can be partially explained by the additional information stored in the switches. Additionally, WWAEs produce comparable classification accuracy to other established frameworks.[17] We will demonstrate that this method provides very clean and informative features.

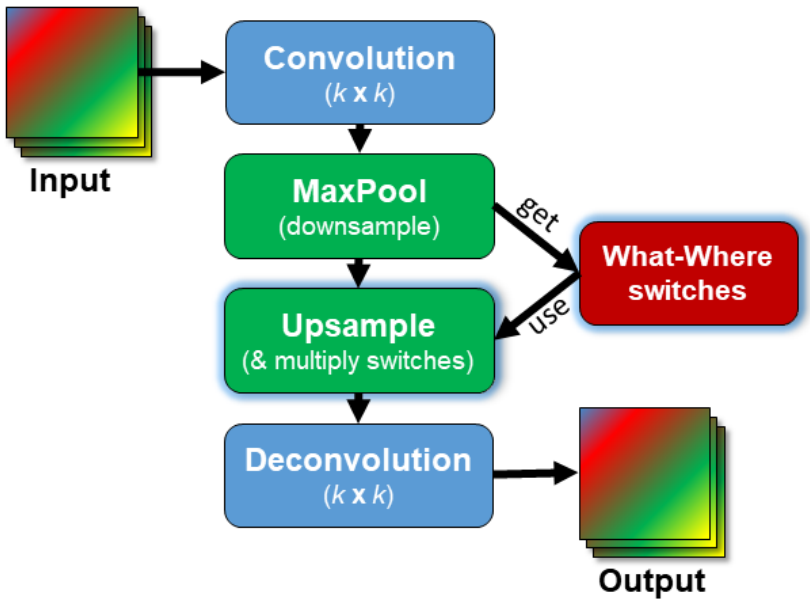

Fig. 1 Diagram of a single-layer convolutional WWAE

Sparsity may be imposed upon the hidden activations of an autoencoder to prevent the pollution of features with irrelevant information, akin to what is used in conventional dictionary learning. Having only a few features strongly activated while all others are negligible conveys meaningful information about the data. This can be achieved by imposing a sparsifying penalty (such as the ℓ_1-norm) to the activations[10,33] or by selecting a certain number of features with the largest activations[31,32]. While an interesting problem, we apply no regularization in this work and leave it as a topic to be explored in the future.

2.2 Collective Feature Extraction

We define *collective features* as groups of primitive features that combine to form more sophisticated patterns. In a more technical sense, it is actually the clustering of feature vectors corresponding to each pixel in the input image. Clustering Gabor feature vectors has seen success by providing a texture-based segmentation of images.[34] Jun et al. performed k-means clustering[35] on deep-level feature representations extracted from a stacked denoising autoencoder.[18] More informative observations may be gleaned from isolating primitive features that naturally occur in close proximity throughout the signal. Sometimes, primitive features alone may not naturally appear to be representative samples of a signal. For example, the features of an image of text should ideally be characters; however, primitive feature extraction may only detect lines and curves. Classifying common groupings of these lines and curves allows for letters to be detected, resulting in more informative features. Moreover, for unlabeled data of unknown origin unlike the text example, the extraction of meaningful and representative features is much more crucial as one may not be familiar with how the primitive features combine.

2.2.1 Segmentation

Recall that during convolution, the filters slide along the signal, and the dot product is taken at each specified position. This results in activation maps corresponding to each filter that detail the presence of the corresponding feature at each particular point. We choose to group primitive features based on these activation maps as they indicate which features are highly active in close proximity. Here, we impose that the stride of the convolution should be one. Therefore, the output of the convolutional layer will be similar in size to the input, and each sample will have unique values in the activation maps and can be classified accordingly. A stride greater than one may be used; however, groups of samples are classified together, which may impede the accuracy. Using strided convolution with no max-pooling or unpooling for segmentation is demonstrated in Section 4.4. The activation maps of the encoding convolutional layer are utilized because it is the location where the feature detectors are trained as opposed to the decoding layer where features are optimized for reconstructing the input.

We begin by performing dilation separately on each activation map. This morphological operation convolves the signal with a kernel B, replacing the image pixel at the anchor point with the maximum value overlapped by B. This operation serves to enlarge bright regions to overpower nearby darker regions; in this application, dilation magnifies regions of heavy activation. The dilated activations are then clustered through the k-means algorithm. These clusters are dubbed the

collective features. A collective feature is technically a distribution, so it cannot be easily visualized, but it should resemble the parts of the input signal where the particular feature is dominant.

2.3 Overcoding

When training a network on a lone signal, we break the signal into windows in order to produce sufficient training and validation data. Once the model is trained, the full signal can be fed through the model to extract the activation maps. However, when clustering the activations of a test image, we noticed some strange artifacts present across the entirety of the image. These effects originate as the filter slides over the edge of the patch; the filter learns only the part of the image that it covers and zeroes elsewhere, creating an edge effect. We propose an *overcoding* method to alleviate this effect.

Our method learns a fragment of the image and reconstructs a smaller portion, ensuring that the filters are entirely contained within that fragment. In Fig. 2, we detail two variants of our proposed method. Note that this method is valid for data in both one and two dimensions; for the 2-D case, simply treat each value as a square.

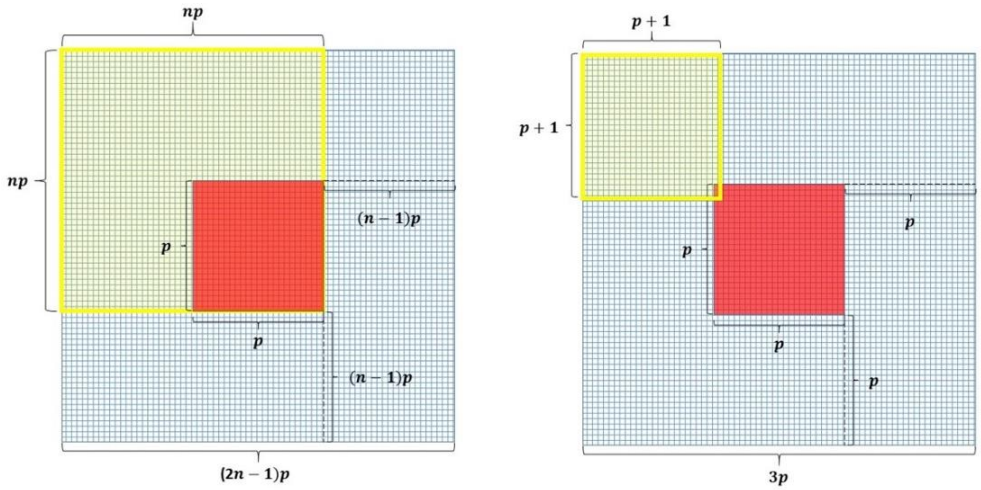

Fig. 2 A diagram of both overcoding processes where the red represents the current patch, the blue grid represents the surrounding image, and the yellow depicts the sliding filter. The left image demonstrates when the kernel size is a factor *n* of the pooling size, while the right image depicts the case when the kernel size is one more than the pooling size.

Let p be the pooling size of the max-pooling layer and $k = np$ be the size of the convolutional kernel for some positive integer n. The image is broken into non-overlapping patches of size $(2n-1)p$, which are fed through a single-layer convolutional WWAE with a stride of 1. After decoding, the reconstructed patches

are cropped to a size of p, shaving off $(n-1)p$ pixels from each edge. We force the convolutional filters to not exceed the edge of the patches; therefore, the predicted p samples will always be entirely covered by the convolutional filter as depicted in Fig. 2.

For our other method, we let $k = p + 1$ be the size of the convolutional kernels and break the image into non-overlapping patches of size $3p$ where the center p samples will be reconstructed as shown previously. In this case, there is one slight addition to the model. The encoding convolution layer will output blocks of size $2p$ since the filters are required to stay fully inside the border of the patches. The max-pooling and unpooling layers leave this size unchanged. In order to have the center p samples reconstructed, we zero pad the processed patches by $\frac{p}{2}$ on all sides; note that these zeroes only impact reconstruction and have negligible impact on the training of the activation maps of the encoding convolutional layer. Additionally, notice that even pooling sizes work much more effectively in this case. The output of the decoding convolutional layer is cropped to the center p samples as shown previously. Unlike the previously detailed case, note that the center p samples may not be entirely covered by the convolutional filters. Figure 2 demonstrates this process.

3. Methods

3.1 Training

We detail the data sets and models used for feature extraction. The general architecture for each type of network is discussed here; refer to each corresponding section for more specific details. Supervised networks with pooling contain a convolutional layer with a stride of one followed by a max-pooling layer, while those with striding contain only a convolutional layer with a stride greater than one. Both of these models possess a layer of dropout before the final decision layer. The unsupervised networks are composed of a single convolutional layer of encoding and decoding. The encoding convolutional layer can possess either a stride of one followed by a max-pooling layer or simply a stride greater than one. For WWAEs, switches are employed to retain relational information on maxima. If pooling is employed, an unpooling layer occurs first during decoding. For WWAEs, the switches restore the maxima to their original location during unpooling. The final decoding convolutional layer will have a stride that matches that of the encoding layer.

3.1.1 MNIST Handwritten Digits Data Set

The MNIST data set[36] of 28 × 28 pixel images of handwritten digits was utilized for feature extraction with the standard training set of 60,000 digits and test set of 10,000 digits, serving to qualitatively compare feature extraction between various forms of supervised and unsupervised single-layer convolutional networks. We performed no further processing of the images themselves. The MNIST digits were deemed too small for the use of overcoding. The hyperparameters for these models are collected in Table 1; note that all models utilize a batch size of 20.

Table 1 Hyperparameters for the neural networks

Type of model	Filters	Kernel size	Pooling/striding size	Epochs	Activations
MNIST supervised	20	14	7	100	ReLu
MNIST unsupervised	20	14	7	100	Linear
Audio no overcoding	40	1024	256	20	Tanh
Audio overcoding $k = p + 1$	40	1024	1023	20	Tanh
Audio overcoding $k = 2p$	40	1024	512	20	Tanh
Audio overcoding $k = 4p$	40	1024	256	20	Tanh
Image no overcoding	50	16	8	450	Linear
Image overcoding $k = p + 1$	50	16	8	450	Linear
Image overcoding $k = 2p$	50	16	15	450	Linear

3.1.2 Audio

Keeping with our theme of sparse data, we utilized a single 30-min and 50-s speech sampled at 22,050 Hz mono.[37] Our objective was to compare the features extracted through various unsupervised means with those extracted from dictionary learning. In the case of max-pooling and striding without overcoding, we determined the ideal features in this case would correspond to phonemes. Using a reasonable estimate that there are two phonemes per second in the English language, we broke the speech into fragments of size 9216 (1024×9). The overcoding cases follow the process outlined in Section 2.2.2 with hyperparameters given in Table 1. During dictionary learning, the speech was broken into patches of size 1024. The algorithm ran for 450 iterations training 70 pairs of encoding and decoding features.

3.1.3 Single Images

We extracted and localized features from single images. Figure 3 depicts the three images upon which our model was separately trained. Our goal was to compare unsupervised methods of feature extraction with that of dictionary learning. In the unsupervised cases without overcoding (max-pooling and striding), the image is broken into patches whose dimensions were equivalent to the pooling size. The overcoding case follows the process of Section 2.2.2. Refer to Table 1 for the hyperparameters of these models. For dictionary learning, the image was broken into 16 × 16 patches. The algorithm ran for 450 iterations training 70 features. As with the neural networks, the algorithm was trained separately on each image.

Lorem ipsum dignis eium fugit aspelluptat eati odis net exeri rectem lia venihil icipsapid qui dolupient quam aceatemque repedi tem lantiae provid quia sitatia temquibuste voluptatquo comnit fugiat invenit fugia quo ditias ipitatat erspel id utesecuptur solorio. Hari veria dis nis et millic tota comnimet inctor sum aut laboressedis deribus illore non et fugitiosam, soluptat in eatur? Endignatem el et ex endicim re occullu ptatem laut audipita num fugit adis delenti cum sus aut iduntur arumqui blam eos molorum quissimint, nis ut aut adisci odic te as everspe ditati accum alit rempel iumque nobis repudamus.

Epro que pra nis atempor ad quis am sim explitem sum et, illaute mposape lestiores rehenis ante senducidus quatet doluptatur sumquia ntiore, et idus volutem volorerio volorem fugit ento blabore pro et, od quam sam re volori aut etur seque velignia voloreic tem es nosam volo magnam que ad que voles et voloribeatem rati ipsae dolentiatur rempore sequunt, quis ea volupti am secuptaquate cus etum eliqui dolorep erferrum diore quibus re omnisci aditibusda qui aut alit, ide et possequias nonsed est int eos il earciis sit aliquidis eum quiam, quam doluptatis vel eum nos pos dolorat emoditiur? Quid que pe veratiae pero molum volorehendit re, unt pa doluptio cus.

Ga. Nam, ommodic aborrorepe plabo. Neque quaestiat hiciae opta inis dolestrum cum, quam ut fugiam, od mosamus aruptaspel modi rerunt paris a volore pressed ut voluptatius voluptae provid quis volo eumquunte exerumq uaernatia serum cus estis autemquis exeris et ut omni consequis essi to qui ne destoreri sinctes placcae aut labo. Ritia dolestiatur autet omnitatus aliciae repudam, sanditatiis doluptatur? Qui apid ut officabo. Et essi nem et debis re dolorio ssinctatem quodipsuntem explique doluptaqui officidempor ad ut volut la doluptam re dolute eatur sequam exerspero explacc ulliquiatio vendustrum etus sed que desseque sum quos voluptatiame core cuptaquibus exerrum et molupta ssiment ressundantem idiciet omnimus re ni berunt prero veliquis non cus.

Aliquibusdae con prernatur aut res dolora aut aut quam comni tem sequo ommoditatet hiliqua tusciae pedit, atures doloritius dolutat emperis et imus dolupta tentibe rspienis niminus ni videbitatem quos re, temquis untibus undaeribus eum faccus, tem erum nem. Entorum illectus del eatur?

Faccuptatur, se nostrum et fugia peria nossunt fuga. Sedicab orectissitae voles evel ipsus ipsae et, sam vit, conseque sit et aspe quo quias se quiandusdae solorumquae vel enimus verumendunt.

Ab into dolora nossequi tem num quunt maxim del intio. Harumenda ne perovit quiant ut alique doluptatia ium sunt volut magnatiis voluptis es enim non earuptae eos doluptatur solorer iorem. Ur rem que dellores nimpos pra sequasped qui quiatqui conem reiunt.

As debit eatiber itatur, aut lat parchilicae pa pellentum ut ident eatia commolo eum nectibusa apel istruntiore idebit eturiatur?

Lupieniendis aut volorerio. Daectot aectestium latem volenimus ut velis alibus ullicils aceaquam derovid elitatet que vel minctume iusam dolupti venis am fugit etus vellit re viducid quiatquiant volestisti torehen tiorestem antis militas nes del ilicaturibus et et est harum, ipsant, natem quos es ipsa velit est, es re volupta temolorum este explant.

Pore vid est, audam facia voluptiae pos ut que nulloria core nihilita istio tem quiscia volore nulliae corpos eatur

Fig. 3 The three images[38–40] that were utilized in this work with resolutions 2868 × 1601 (office), 5430 × 3620 (parking lot), and 1425 × 1030 (text)

3.2 Segmentation

We compared the efficacy of the segmentation algorithm on the models trained on single images. A 24 × 24 filter was used for dilation with the center as the anchor point; it is depicted in Fig. 4. The activations of the text image were grouped into 30 clusters, while those of the office and parking lot images were classified into 15 clusters.

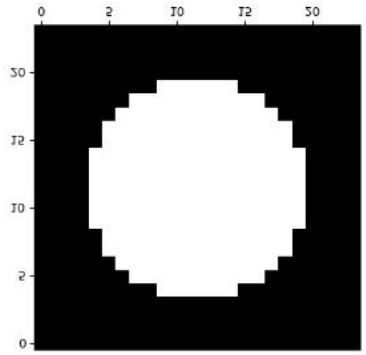

Fig. 4 Filter utilized for dilation during segmentation

4. Results and Discussion

Clear and descriptive features can be vital in understanding unlabeled data. Here, we empirically compare features derived from our method with those of similar methods. Additionally, we demonstrate how our feature extraction method works in tandem with feature segmentation.

4.1 MNIST

We compared our method of feature extraction with various supervised and unsupervised neural networks on the MNIST data set as benchmarks. Figure 5 displays the features of each model. Note that since there are an ample number of digits in the MNIST data set and the dimensions of the digits themselves are only 28 pixels, no overcoding was required in any of the models. We expect supervised neural networks to possess generally clean features, as the learning process is guided by the labels provided. This is demonstrated by the lines and curves in Figs. 5a and 5b. However, the presence of multiple curves in a single feature indicates that there is some entanglement occurring. These models do a comparatively better job than the unsupervised models that do not use what-where switches. As depicted in Figs. 5c and 5d, simply striding or max-pooling without what-where switches in the unsupervised case produces an unintelligible blur of high-frequency components. However, in our model containing the what-where method, extremely clean, untangled “wavelets” are extracted. These features are uniquely descriptive because one can easily determine how each digit can be reconstructed from them.

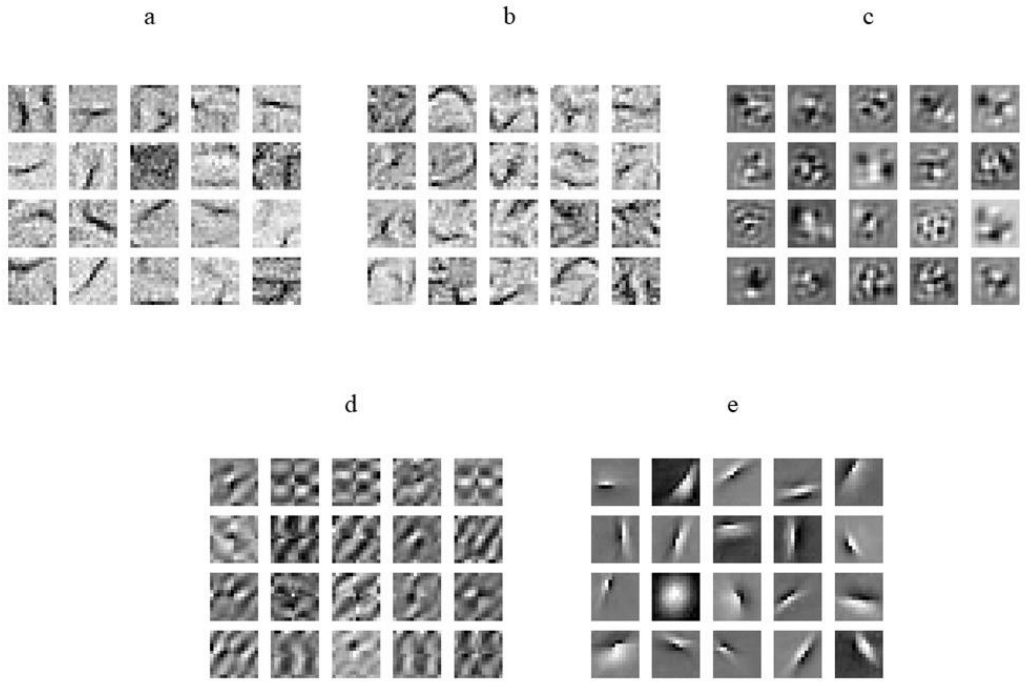

Fig. 5 Features extracted from the various models trained on the MNIST data: a) supervised striding, b) supervised max-pooling, c) unsupervised striding, d) unsupervised max-pooling without what-where switches, and e) what-where switches

One may wonder how an unsupervised model behaves on similar inputs, such as two of the same digits in our case. In theory, the model should utilize similar features to reconstruct the digits—that is, the decoding activation maps are expected to be similar. Recall that these models are unsupervised, so the labels of the data are never processed. While this is not necessary for reconstruction accuracy, it is interesting because it can be roughly viewed as unsupervised segmentation. The activation maps can be visualized through dimensionality reduction techniques. Here, we employ the t-SNE algorithm,[41] which converts pairs of the data to a probability distribution where similarity of the data in the pairs correlates to the probability of selection. A similar distribution is defined in the lower dimension such that sum of the Kullback-Leibler divergences[42] over all data points is minimized. Thus, similar high-dimensional data points will be positioned closely in the lower-dimensional representation.

We fed the 10,000 test digits to each trained unsupervised model and visualized the decoding activation maps in two dimensions using t-SNE. The results of this are presented in Fig. 6. Note that while the colors of the figure are derived from the labels of the digits, the models had no access to these labels. The WWAE provided the cleanest segmentation of the three unsupervised models. The other two models consistently confused the digits 4 and 9 and had some trouble distinguishing the

digits 3, 5, and 8. Notice that the WWAE was not perfect as a small subset of the fives was mistaken for the digit 3. Despite this, the WWAE was able to cleanly segment the 10 digits without access to labels. This is an extremely desirable property regarding unlabeled data, indicating that the visualization of decoding activation maps can inherently categorize the data. It is interesting to point out that the results of Turchenko et al.[22] demonstrate that WWAEs provided the least accurate segmentation of the five unsupervised models. We attribute the difference in our results to the use of a shallow model with a much larger pooling size—Turchenko et al. utilized a two-layer model with a pooling size of two.

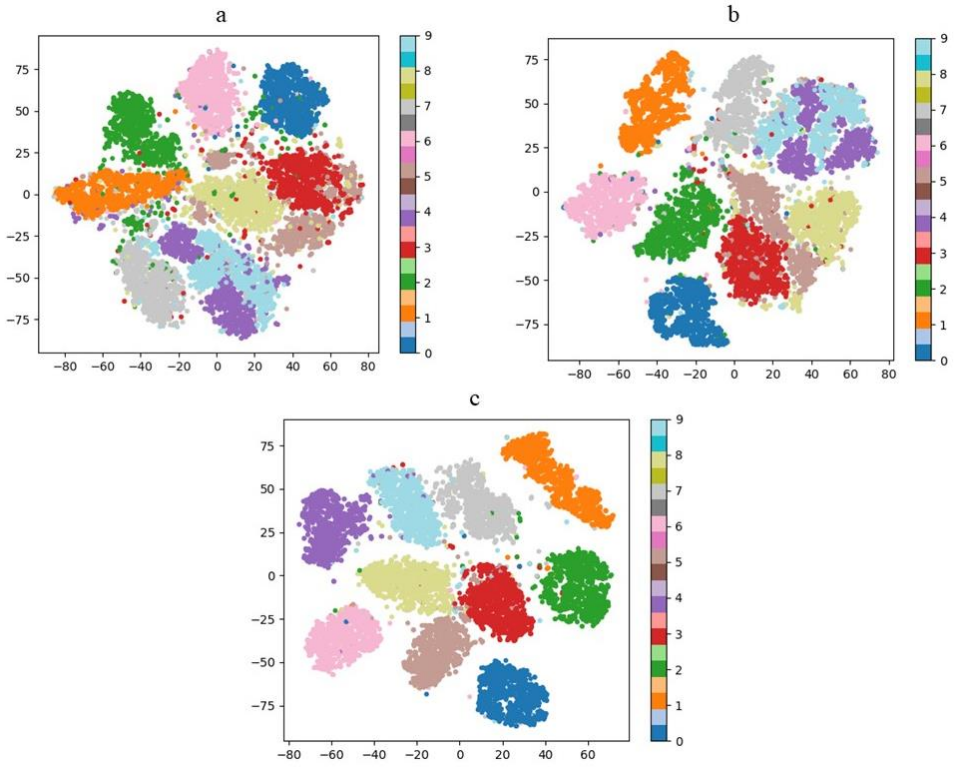

Fig. 6 Results of dimensionality reduction of the activation maps for the unsupervised models: a) striding, max-pooling; b) without what-where switches; and c) with what-where switches

4.2 Audio

We utilized human speech data to demonstrate the capabilities of handling 1-D data; the results are given in Fig. 7. In this instance, we employed dictionary learning for comparison. However, without any shift invariance, many of the features display a similar oscillatory behavior with little variation. The striding and pooling without what-where models both produce very homogenous features. The features from striding are flat on the ends with a small region of activity in the middle, while those from pooling with no what-where generally follow the same

indistinct pattern. The model that incorporates what-where produces a variety of clean features, representing many distinct waveforms. This is expected as the what-where method introduces shift-invariance. We implemented three versions of overcoding since the single speech was broken into patches during training. The models with overcoding generally produced the largest variety of distinct features, indicating a more accurate depiction of characteristics of the data. It is much more difficult to visually interpret 1-D data; thus, having a wider range of visually distinct features offers a more viable survey of the data set. Notice, however, that there is some sort of edge effect present in a few of the overcoding features.

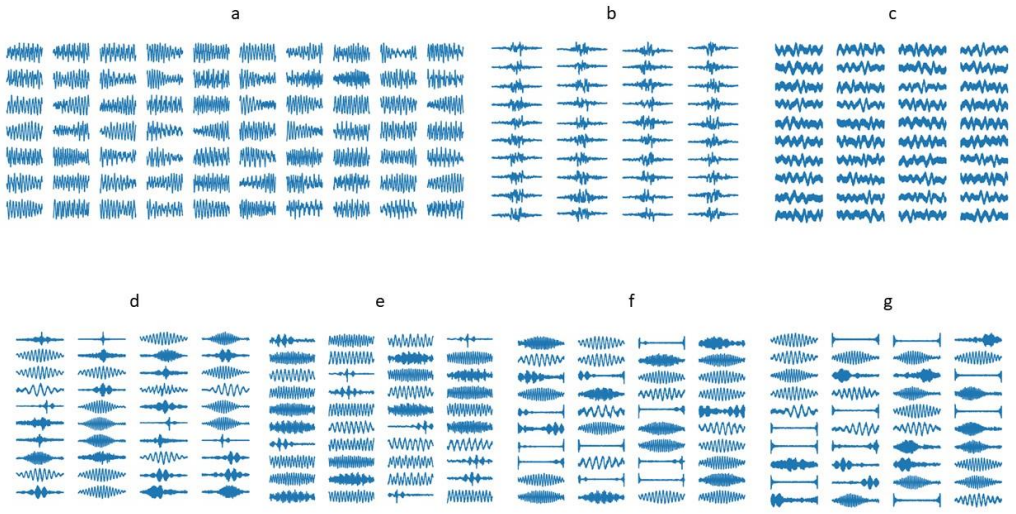

Fig. 7 Features from various unsupervised models trained on the audio data: a) dictionary learning, b) striding, c) max-pooling without what-where switches, d) what-where switches, e) overcoding with k = p+1, f) overcoding with k = 2p, and g) overcoding with k = 4p

4.3 Single Images

Single images served as 2-D examples for the comparison of dictionary learning and other unsupervised neural networks. A picture of a page of text was selected for the simplicity and repetitiveness of the characters. The black, repeated text on a white background provides optimal conditions for descriptive features. Additionally, we know that lines, curves, and parts of letters are the ideal result; this gives a reasonable benchmark for our comparisons. Figure 8 displays the extracted features. Like both cases above, the features extracted with unsupervised striding are incoherent. Because the image has fairly high resolution, the what-where method without overcoding struggles to capture clear features; while some distinct lines are present, many of the features are tangled together. Dictionary learning does a fairly decent job at extracting portions of letters. However, many letters are jumbled together in a single feature, failing to provide a clean, descriptive summary. The features extracted from the what-where method present either clean

curves or sections of letters with very few being tangled. It is auspicious that legible letters can be extracted.

Fig. 8 Features extracted from various unsupervised models trained on the text image: a) dictionary, b) striding, c) what-where switches without overcoding, d) overcoding with k = p+1, and e) overcoding with k = 2p

The other two single images depicted much more complicated scenes, presenting a more difficult challenge for the networks. Figure 9 displays the respective features of the office and parking lot pictures. Notice that while the features from dictionary learning for both pictures contain some clean curves, many of the features are just a saturated color or completely noise. As we have demonstrated before, the features obtained from simply striding are unintuitive jumbles. For these images, utilizing what-where without any overcoding produces tangled amalgamations of different features, likely due to the complexity of the images. While a few of the features obtained from overcoding are single colors, the overcoding method provides the cleanest features. Clear curves can be detected in the bulk of the features. However, unlike the previous cases, the features cannot be easily connected to the original signal.

Fig. 9 Features extracted from various unsupervised models trained on the office and parking lot images respectively: a) dictionary, b) striding, c) what-where switches without overcoding, d) overcoding with k = p+1, and e) overcoding with k = 2p

The other two single images depicted much more complicated scenes, presenting a more difficult challenge for the networks. Figure 9 displays the respective features of the office and parking lot pictures. Notice that while the features from dictionary learning for both pictures contain some clean curves, many of the features are just a saturated color or completely noise. As we have demonstrated before, the features obtained from simply striding are unintuitive jumbles. For these images, utilizing what-where without any overcoding produces tangled amalgamations of different features, likely due to the complexity of the images. While a few of the features obtained from overcoding are single colors, the overcoding method provides the cleanest features. Clear curves can be detected in most of the features. Unlike the previous cases, the features cannot be easily connected to the original signal.

4.4 Segmentation

The segmentation technique is intended to determine which features commonly occur together in order to classify features that may not be extracted by existing methods. We focus on the text image first as it provides the most poignant results. Figure 10 contains just the first paragraph from the text; the entire results for each method are contained in the supplemental materials.

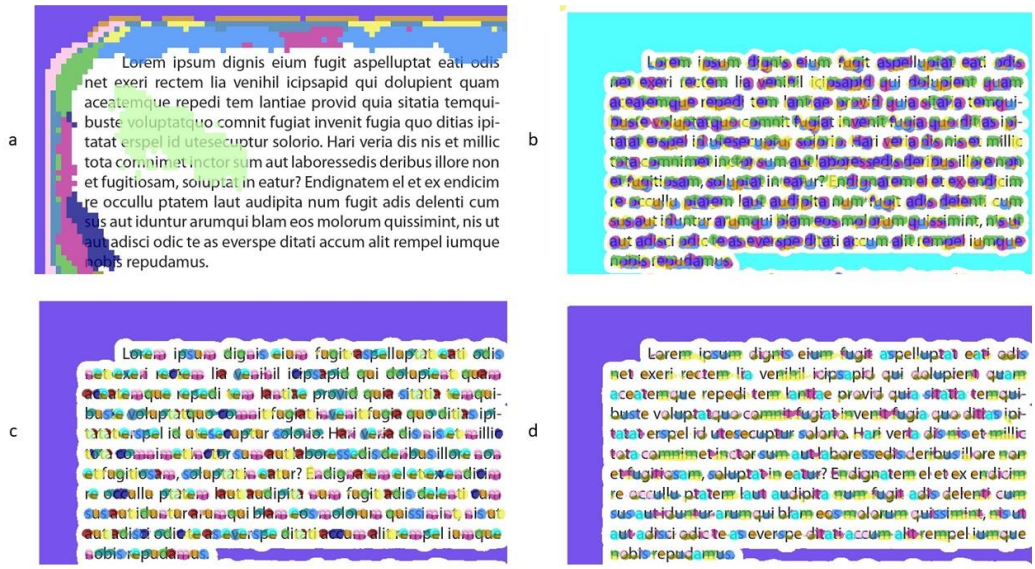

Fig. 10 Results of segmentation for various unsupervised neural networks trained on the text image: a) striding, b) what-where switches without overcoding, c) overcoding with k = p+1, and d) overcoding with k = 2p

The striding method is unable to detect any letters, likely due to the incomprehensible features. While the method of what-where without overcoding does capture letters, notice that no color is unique to a specific letter. This does a decent job of detecting letters in general but is not necessarily reliable in segmenting particular letters. However, the methods that utilize overcoding actually can extract single letters. Notice that in the case where $k = p + 1$, the letters *v*, *s*, *c*, and *u* are detected by a single feature while the letter *e* is captured by a combination of two features. The letter *t* is completely captured by one of two features depending on what character follows; the same can be said for the letter *a* depending on what character precedes it. The letters *m*, *n*, and *h* are detected by a set of two features. We now focus on the case where $k = 2p$. Here, the letters *a*, *e*, *s*, and *u* are all detectable with a unique collective feature. Unlike the previous case, the other features capture sections of many letters as opposed to individual letters themselves. The cleaner features extracted in Section 4.3 result in a much more informative collective feature extraction.

Figure 11 displays the results from the segmentation method on the other two images. Since these scenes are much more complicated than a picture of text, the results are not as descriptive as the previous case. Considering the complexity of the scenes and difficulty of the task, the results are quite promising. Notice that for these images, the what-where method with no overcoding provided the least accurate segmentation since homogenous regions are often broken into multiple regions. While striding does a better job in this case, there are some egregious flaws with the segmentation. In the parking lot image, the filter activated by the trees is also present on a few of the cars while portions of the door frame share other features in the office scene. This is the only case where the overcoding did not provide a considerable improvement over the other methods. The wooden texture of the bookcase, table, and doors of the office are almost entirely captured by a single feature in the $k = 2p$ case. Additionally, in the $k = p + 1$ case, most of the skin is represented by one feature. For the parking lot picture, the bricks, taillights, and pavement are each almost entirely detected by a single feature in both cases. While the segmentation does appear to be much cleaner than that of the other two methods, there are still a few glaring flaws with the activations of the overcoding method. Despite this, the accuracy demonstrated is remarkable given that the autoencoder was trained to reconstruct a single image.

Fig. 11 Results of segmentation for various unsupervised neural networks trained on the office and parking lot images respectively: a) striding, b) what-where switches without overcoding, c) overcoding with k = p+1, and d) overcoding with k = 2p

5. Conclusion

In this work, we show how a neural network architecture originally developed for pre-training a convolutional neural network can be used to extract shift-invariant features suitable for partial segmentation of images. Unlike traditional dictionary learning and striding-based autoencoders, shift-invariance greatly improves the legibility of the features. Clustering of the feature prevalence in the signal (image), leads to collective features suitable for unsupervised segmentation. Overcoding removes edge artifacts that often plague autoencoders. Given the ready availability of various open source neural network frameworks, such as Tensorflow and PyTorch, the shallow WWAE is an accessible way to perform shift-invariant dictionary learning. Future research in this area might entail evaluating the utility of this algorithm for developing compressed representations.

6. References

1. Caelles S, Maninis K-K, Pont-Tuset J, Leal-Taixé L, Cremers D, Van Gool L. One-shot video object segmentation. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR); 2017 July 21–26; Honolulu, HI. Los Alamitos (CA): IEEE Computer Society. p. 221–230.

2. Fei-Fei L, Fergus R, Perona P. One-shot learning of object categories. IEEE Trans Pattern Anal Mach Intell. 2006;28(4):594–611.

3. Long J, Shelhamer E, Darrell T. Fully convolutional networks for semantic segmentation. IEEE Trans Pattern Anal Mach Intell. 2016;79(10):1337–1342.

4. Hu J, Tan Y-P. Nonlinear dictionary learning with application to image classification. Pattern Recognit. 2018;75:282–291.

5. Lin G, Yang M, Yang J, Shen L, Xie W. Robust, discriminative and comprehensive dictionary learning for face recognition. Pattern Recognit. 2018;81:341–356.

6. Chen Y, Su J. Sparse embedded dictionary learning on face recognition. Pattern Recognit. 2017;64:51–59.

7. Zhang G, Sun H, Ji Z, Yuan Y-H, Sun Q. Cost-sensitive dictionary learning for face recognition. Pattern Recognit. 2016;60:613–629.

8. Ramírez I, Sprechmann P, Sapiro G. Classification and clustering via dictionary learning with structured incoherence and shared features. In: 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR); 2010 June 13–18; San Francisco, CA. Los Alamitos (CA): IEEE Computer Society. p. 3501–3508.

9. Zolhavarieh S, Aghabozorgi S, Wah Teh Y. A review of subsequence time series clustering. Sci World J. 2014. Article ID 312521.

10. Ranzato MA, Huang F-J, Boureau Y-L, LeCun Y. Unsupervised learning of invariant feature hierarchies with applications to object recognition. In: 2007 IEEE Conference on Computer Vision and Pattern Recognition (CVPR); 2007 June 17–22; Minneapolis, MN. Los Alamitos (CA): IEEE Computer Society. p. 1–8.

11. Mailhé B, Lesage S, Gribonval R, Bimbot F, Vandergheynst P. Shift-invariant dictionary learning for sparse representations: extending K-SVD. In: 2008 European Signal Processing Conference (EUSIPCO); 2008 Aug 25–29; Lausanne, Switzerland. Piscataway (NJ): IEEE. p. 5-p.

12. Pope G, Aubel C, Studer C. Learning phase-invariant dictionaries. In: 2013 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP); 2013 May 26–31; Vancouver, Canada. p. 5979–5983.

13. Gowreesunker BV, Tewfik A. A shift tolerant dictionary training method. Proceedings of 2009 Signal Processing with Adaptive Sparse Structured Representations (SPARS); 2009 Apr 6–9; St Malo, France.

14. Thiagarajan JJ, Ramamurthy KN, Spanias A. Shift-invariant sparse representation of images using learned dictionaries. In: 2008 IEEE Workshop on Machine Learning for Signal Processing (MLSP); 2008 Oct 16–19; Cancun, Mexico. Piscataway (NJ): IEEE. p. 145–150.

15. Aharon M, Elad M, Bruckstein A. K-SVD: an algorithm for designing overcomplete dictionaries for sparse representation. IEEE Trans Signal Process. 2006;54(11):4311–22.

16. Gowreesunker BV, Tewfik AH. A novel subspace clustering method for dictionary design. In: Adali T, Jutten C, Travassos Romano JM, Barros AK, editors. ICA 2009. Independent Component Analysis and Signal Separation, 8th International Conference; 2009 Mar; Paraty, Brazil. Berlin (Germany): Springer. p. 34–41.

17. Zhao J, Mathieu M, Goroshin R, LeCun Y. Stacked what-where auto-encoders. Ithaca (NY): Cornell University; 2016 Feb 14 [accessed 2019 Mar 1]. https://arxiv.org/abs/1506.02351.

18. Yu J, Huang D, Wei Z. Unsupervised image segmentation via stacked denoising auto-encoder and hierarchical patch indexing. Signal Process. 2018;143:346–353.

19. Vincent P, Larochelle H, Lajoie I, Bengio Y, Manzagol P-A. Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion. J Mach Learn Res. 2010;11:3371–3408.

20. Alex V, Vaidhya K, Thirunavukkarasu S, Kesavdas C, Krishnamurthi G. Semi-supervised learning using denoising auto-encoders for brain lesion detection and segmentation. Ithaca (NY): Cornell University; 2017 Jan 10 [accessed 2019 Mar 1]. https://arxiv.org/abs/1611.08664.

21. Shin H-C, Orton MR, Collins DJ, Doran SJ, Leach MO. Stacked autoencoders for unsupervised feature learning and multiple organ detection in a pilot study using 4D patient data. IEEE Trans Pattern Anal Mach Intell. 2013;35(8):1930–1943.

22. Turchenko V, Chalmers E, Luczak A. A deep convolutional auto-encoder with pooling-unpooling layers in Caffe. Ithaca (NY): Cornell University; 2017 Jan 18 [accessed 2019 Mar 1]. https://arxiv.org/abs/1701.04949.

23. Girshick R, Donahue J, Darrell T, Malik J. Rich feature hierarchies for accurate object detection and semantic segmentation. In: 2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR); 2014 June 23–28; Columbus, Ohio. Piscataway (NJ): IEEE. p. 580–587.

24. Donoho DL. For most large underdetermined systems of linear equations the minimal $\ell 1$-norm solution is also the sparsest solution. Commun Pur Appl Math. 2006;59(6):797–829.

25. Pati YC, Rezaiifar R, Krishnaprasad PS. Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition. Proceedings of the 27th Asilomar Conference on Signals, Systems and Computers; 1993 Nov 1–3; Pacific Grove, CA. Piscataway (NJ): IEEE. p. 40–44.

26. Gorodnitsky IF, Rao BD. Sparse signal reconstruction from limited data using FOCUSS: a re-weighted minimum norm algorithm. IEEE Trans Signal Process. 1997;45(3):600–616.

27. Engan K, Aase SO, Husoy JH. Method of optimal directions for frame design. In: 1999 IEEE International Conference on Acoustics, Speech, and Signal Processing; 1999 Mar 15–19; Phoenix, AZ. Piscataway (NJ): IEEE. p. 2443–2446.

28. Zheng G, Yang Y, Carbonell J. Efficient shift-invariant dictionary learning. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining; 2016 Aug 13–17; San Francisco, CA. New York (NY): ACM. p. 2095–2104.

29. Lee H, Battle A, Raina R, Ng AY. Efficient sparse coding algorithms. Proceedings of the 21st Annual Conference on Neural Information Processing Systems (NIPS); 2007; Vancouver, Canada. p. 801–808.

30. Springenberg JT, Dosovitskiy A, Brox T, Riedmiller M. Striving for simplicity: the all convolutional net. Ithaca (NY): Cornell University; 2015 Apr 13 [accessed 2019 Mar 1]. https://arxiv.org/abs/1412.6806.

31. Makhzani A, Frey B. K-sparse autoencoders. Ithaca (NY): Cornell University; 2014 Mar 22 [accessed 2019 Mar 1]. https://arxiv.org/abs/1312.5663.

32. Makhzani A, Frey B. Winner-take-all autoencoders. In: 2015 Advances in Neural Information Processing Systems (NIPS); 2015 Dec 7–12; Montreal, Canada. p. 2791–2799.

33. Poultney C, Ranzato MA, Chopra S, LeCun Y. Efficient learning of sparse representations with an energy-based model. In: 2007 Advances in Neural Information Processing Systems (NIPS); 2007; Vancouver, Canada. p. 1137–1144.

34. Lu S-Y, Hernandez JE, Clark GA. Texture segmentation by clustering of Gabor feature vectors. In: International Joint Conference on Neural Networks (IJCNN); 1991 July 8–12; Seattle, WA. New York (NY): IEEE. p. 683–688.

35. Lloyd S. Least squares quantization in PCM. IEEE Trans on Inform Theory. 1982;28(2):129–137.

36. LeCun Y. The MNIST database of handwritten digits; 1998 [accessed 2019 Mar 1]. http://yann.lecun.com/exdb/mnist.

37. American Rhetoric. Online speech bank; 2017 Nov 30 [accessed 2019 Mar 1]. https://www.americanrhetoric.com/speeches/wariniraq/davidpetraeusoniraq .htm.

38. PIXNIO; 2018 [accessed 2019 Mar 1]. https://pixnio.com/people/seven-people-at-meeting-on-office.

39. Wikimedia Commons; 2018 Mar 20 [accessed 2019 Mar 1]. https://commons.wikimedia.org/wiki/File:Ugly_Parking_Lot_(15838489529) .jpg.

40. Lorem Ipsum Example; 2017 [accessed 2019 Mar 7]. http://thelawlers.com/Blognosticator/wp-content/uploads/2017/02/Lorem-Ipsum-example-1024x740.jpg.

41. van der Maaten L, Hinton G. Visualizing data using t-SNE. J Mach Learn Res. 2008;9:2579–2605.

42. Kullback S, Leibler RA. On information and sufficiency. Ann Math Stat. 1951;22(1):79–86.

List of Symbols, Abbreviations, and Acronyms

1-D	1-dimensional
2-D	2-dimensional
DOD	US Department of Defense
FOCUSS	Focal Underdetermined System Solver
K-SVD	k-means singular value decomposition
MNIST	Modified National Institute of Standards and Technology
MOD	Method of Optimal Directions
OMP	Orthogonal Matching Pursuit
WWAE	what-where autoencoder

1 (PDF) DEFENSE TECHNICAL INFORMATION CTR DTIC OCA

2 (PDF) CCDC ARL
IMAL HRA
RECORDS MGMT
FCDD DCL
TECH LIB

1 (PDF) GOVT PRINTG OFC
A MALHOTRA

6 (PDF) ARL
FCDD RLC
T PHAM
FCDD RLC HS
D SHIRES
M VINDIOLA
FCDD RLC HC
E CHIN
FCDD RLC I
S YOUNG
FCDD RLC S
R VALISETTY