From sBoW to dCoT Marginalized Encoders for Text Representation [*]

Zhixiang (Eddie) Xu
Washington University in St. Louis
One Brookings Dr.
St. Louis, MO, USA
zhixiang.xu@wustl.edu

Minmin Chen
Washington University in St. Louis
One Brookings Dr.
St. Louis, MO, USA
chenm@wustl.edu

Kilian Q. Weinberger
Washington University in St. Louis
One Brookings Dr.
St. Louis, MO, USA
kilian@wustl.edu

Fei Sha
University of Southern California
941 West 37th Place
Los Angeles, CA, USA
feisha@usc.edu

ABSTRACT

In text mining, information retrieval, and machine learning, text documents are commonly represented through variants of *sparse Bag of Words* (sBoW) vectors (*e.g.* TF-IDF [1]). Although simple and intuitive, sBoW style representations suffer from their inherent over-sparsity and fail to capture word-level synonymy and polysemy. Especially when labeled data is limited (*e.g.* in document classification), or the text documents are short (*e.g.* emails or abstracts), many features are rarely observed within the training corpus. This leads to overfitting and reduced generalization accuracy. In this paper we propose *Dense Cohort of Terms* (dCoT), an unsupervised algorithm to learn improved sBoW document features. dCoT explicitly models absent words by removing and reconstructing random sub-sets of words in the unlabeled corpus. With this approach, dCoT learns to reconstruct frequent words from co-occurring infrequent words and maps the high dimensional sparse sBoW vectors into a low-dimensional dense representation. We show that the feature removal can be marginalized out and that the reconstruction can be solved for in closed-form. We demonstrate empirically, on several benchmark datasets, that dCoT features significantly improve the classification accuracy across several document classification tasks.

[*]A full version of this paper is available at `http://www.cse.wustl.edu/~xuzx/research/publications/`

CIKM'12, October 29–November 2, 2012, Maui, HI, USA.
Copyright 2012 ACM 978-1-4503-1156-4/12/10 ...$15.00.

Categories and Subject Descriptors

H.3 [**Information Storage and Retrieval**]: Miscellaneous; I.5.2 [**Pattern Recognition**]: Design Methodology—*Feature evaluation and selection*

General Terms

Machine Learning for IR

Keywords

Denoising Autoencoder, Marginalized, Stacked, Text features

1. INTRODUCTION

The feature representation of text documents plays a critical role in many applications of data-mining and information retrieval. The *sparse Bag of Words* (sBoW) representation is arguably one of the most commonly used and effective approaches. Each document is represented by a high dimensional sparse vector, where each dimension corresponds to the term frequency of a unique word within a dictionary or hash-table [2]. A natural extension is TF-IDF [1], where the term frequency counts are discounted by the inverse-document-frequencies. Despite its wide-spread use with text and image data [3], sBoW does have some severe limitations, mainly due to its often excessive sparsity.

Although the Oxford English Dictionary contains approximately $600,000$ unique words, it is fair to say that the essence of most written text can be expressed with a far smaller vocabulary (*e.g.* 5000–10000 unique words). For example the words *splendid, spectacular, terrific, glorious, resplendent* are all to some degree synonymous with the word *good.* However, as sBoW does not capture synonymy, a document that uses "splendid" will be considered dissimilar from a document that uses the word "terrific". A classifier, trained to predict the sentiment of a document, would have to be exposed to a very large set of labeled examples to learn that all these words are predictive towards a positive sentiment.

In this paper, we propose a novel feature learning algorithm that directly addresses the problems of excessive sparsity in sBoW representations. Our algorithm, which we refer to as Dense Cohort of Terms (dCoT), maps high-dimensional (overly) *sparse* vectors into a low-dimensional *dense* representation. The mapping is trained to reconstruct frequent from *in*frequent words. The training process is entirely unsupervised, as we generate training instances by randomly and repeatedly removing common words from text documents. These removed words are then reconstructed from the remaining text. In this paper we show that the feature removal process can be marginalized out and the reconstruction can be solved for in closed form. The resulting algorithm is a closed-form transformation of the original sBoW features, which is extremely fast to train (on the order of seconds) and apply (milliseconds).

Our empirical results indicate that dCoT is useful for several reasons. First, it provides researchers with an efficient and convenient method to learn better feature representation for sBoW documents, and can be used in a large variety of data-mining, learning and retrieval tasks. Second, we demonstrate that it clearly outperforms existing document representations [1, 4, 5] on several classification tasks. Finally, it is much faster than most competing algorithms.

2. RELATED WORK

Over the years, a great number of models have been developed to describe textual corpora, including vector space models [6, 7, 8, 9, 10], and topic models [4, 11, 12, 13]. Vector space models reduce each document in the corpus to a vector of real numbers, each of which reflects the counts of an unordered collection of words. Among them, the most popular one is the TF-IDF scheme [8], where each dimension of the feature vector computes the term frequency count factored by the inverse document frequency count. By downweighting terms that are common in the entire corpus, it effectively identifies a subset of terms that are discriminative for documents in the corpus. Though simple and efficient, TF-IDF reveals little of the correlations between terms, thus fails to capture some basic linguistic notions such as synonymy and polysemy. Latent Semantic Index (LSI) [5] attempts to overcome this. It applies Singular Value Decomposition (SVD) [14] to the TF-IDF (or sBoW) features to find a so-called latent semantic space that retains most of the variances in the corpus. Each feature in the new space is a linear combination of the original TF-IDF features, which naturally handles the synonymy problem.

Topic modeling develops generative statistical models to discover the hidden "topic" that occur in the corpus. Probabilistic LSI [11], which is proposed as an alternative to LSI, models each document as a mixture of a fixed set of topics, and each word as a sample generated from a single topic. The limitation of probabilistic LSI is that the mixture of topics is modeled explicitly for each training data using a large set of individual parameters, hence, there is no natural way to assign probabilities to unseen documents. Latent Dirichlet Allocation (LDA) [4] solves the problem by introducing a Dirichlet prior on the topic distribution, and treating the mixing weights as multinomial distributed random variables. It is probably the most commonly used topic models nowadays, and the posterior Dirichlet parameters are often used as the low dimensional representation for various tasks [4]. [15] use non-linear dimensionality reduction [16] to embed text data into a low dimensional space, while preserving pair-wise distances between documents. It is fair to say that their approach is computationally most demanding. Similarly to LSI, pLSI and LDA, our algorithm also maps the sparse sBoW features into a low dimensional dense representation. However it is faster to train and addresses the problem of synonymy more explicitly.

3. dCoT

First, we introduce notations that will be used throughout the paper. Let $D = \{\mathbf{w}_1, \cdots, \mathbf{w}_d\}$ be the dictionary of words that appear in the text corpus, with size $d = |D|$. Each input document is represented as a vector $\mathbf{x} \in \mathcal{R}^d$, where each dimension x_j counts the appearance of word $\mathbf{w}_j$ in this document. Let $\mathbf{X} = [\mathbf{x}_1, \cdots, \mathbf{x}_n]$ denote the corpus. Assume that the first $n_l \ll n$ documents are accompanied by corresponding labels $\{y_1, \cdots, y_{n_l}\} \in \mathcal{Y}$, drawn from some joint distribution $\mathcal{D}$.

In this section we introduce the algorithm dCoT, which translates sparse sBoW vectors $\mathbf{x} \in \mathcal{R}^d$ into denser and lower dimensional prototype vectors. We first define the concept of *prototype* terms and then derive the algorithm step-by-step.

Prototype features. Let $P = \{\mathbf{w}_{p_1}, \cdots, \mathbf{w}_{p_r}\} \subset D$, with $|P| = r$ and $r \ll d$, denote a strict subset of the vocabulary D, which we refer to as *prototype* terms. Our algorithm aims to "translate" each term in D into one or more of these prototype words with similar meaning. Several choices are possible to identify P, but a typical heuristic is to pick the r most frequent terms in D. The most frequent terms can be thought of as representative expressions for sets of synonyms — *e.g.* the frequent word *good* represents the rare words *splendid, spectacular, terrific, glorious.* For this choice of P, dCoT translates *rare* words into *frequent* words.

Corruption. The goal of dCoT is to learn a mapping $\mathbf{W} : \mathcal{R}^d \to \mathcal{R}^r$, which "translates" the original sBoW vectors in $\mathcal{R}^d$ into a combination of prototype terms in $\mathcal{R}^r$. Our training of $\mathbf{W}$ is based on one crucial insight: If a prototype term already exists in some input $\mathbf{x}$, $\mathbf{W}$ should be able to predict it from the remaining terms in $\mathbf{x}$. We therefore artificially create a supervised dataset from unlabeled data by *removing* (*i.e.* setting to zero) each term in $\mathbf{x}$ with some probability $(1-p)$. We perform this m times and refer to the resulting corrupted vectors as $\hat{\mathbf{x}}^1, \ldots, \hat{\mathbf{x}}^m$. We not only remove prototype features but all features, to generate more diverse input samples. (In the subsequent section we will show that in fact we never actually have to create this corrupted dataset, as its creation can be marginalized out entirely — but for now let us pretend it actually exists.)

Reconstruction. In addition to the corruptions, for each input $\mathbf{x}_i$ we create a sub-vector $\bar{\mathbf{x}}_i = [x_{p_1}, \cdots, x_{p_r}]^\top \in \mathcal{R}^r$ which only contains its prototype features. A mapping $\mathbf{W} \in \mathcal{R}^{r\times d}$ is then learned to reconstructs the prototype features from the corrupted version $\hat{x}_i$, by minimizing the squared reconstruction error,

$$\frac{1}{2nm} \sum_{i=1}^{n} \sum_{j=1}^{m} \|\bar{\mathbf{x}}_i - \mathbf{W}\hat{\mathbf{x}}_i^j\|^2. \tag{1}$$

To simplify notation, we assume that a constant feature is added to the corrupted input, $\hat{\mathbf{x}}_i = [\hat{\mathbf{x}}_i; 1]$, and an appropriate bias is incorporated within the mapping $\mathbf{W} = [\mathbf{W}, \mathbf{b}]$. Note that the constant feature is never corrupted. The bias term has the important task of reconstructing the average occurrence of the prototype features.

Let us define a design matrix

$$\overline{\mathbf{X}} = [\underbrace{\bar{\mathbf{x}}_1, \cdots, \bar{\mathbf{x}}_1}_{m}, \cdots, \underbrace{\bar{\mathbf{x}}_n, \cdots, \bar{\mathbf{x}}_n}_{m}] \in \mathcal{R}^{r\times nm}$$

as the m copies of the prototype features of the inputs. Similarly, we denote the m corruptions of the original inputs as $\widehat{\mathbf{X}} = [\underbrace{\hat{\mathbf{x}}_1^1, \cdots, \hat{\mathbf{x}}_1^m}_{m}, \cdots, \underbrace{\hat{\mathbf{x}}_n^1, \cdots, \hat{\mathbf{x}}_n^m}_{m}] \in \mathcal{R}^{d\times nm}$. With this notation, the loss in eq. (1) reduces to

$$\frac{1}{2nm}\|\overline{\mathbf{X}} - \mathbf{W}\widehat{\mathbf{X}}\|_F^2, \tag{2}$$

where $\|\cdot\|_F^2$ denotes the squared Frobenius norm. The solution to (2) can be obtained under closed-form as the solution to the well-known ordinary least square.

$$\mathbf{W} = \mathbf{R}\mathbf{Q}^{-1} \text{ with } \mathbf{Q} = \widehat{\mathbf{X}}\widehat{\mathbf{X}}^\top \text{ and } \mathbf{R} = \overline{\mathbf{X}}\widehat{\mathbf{X}}^\top. \tag{3}$$

Marginalized corruption. Ideally, we would like the number of corrupted versions become very large, *i.e.* $m \to \infty$. By the weak law of large numbers, $\mathbf{R}$ and $\mathbf{Q}$ then converge to their expectations and (3) becomes

$$\mathbf{W} = E[\mathbf{R}]E[\mathbf{Q}]^{-1}, \tag{4}$$

with the expectations of $\mathbf{R}$ and $\mathbf{Q}$ defined as

$$E[\mathbf{Q}] = \sum_{i=1}^{n} E[\hat{\mathbf{x}}_i\hat{\mathbf{x}}_i^\top], \; E[\mathbf{R}] = \sum_{i=1}^{n} E[\bar{\mathbf{x}}_i\hat{\mathbf{x}}_i^\top]. \tag{5}$$

The uniform corruption allows us to compute the expectations in (5) in closed form. Let us define a vector $q = [p, \dots, p, 1]^\top \in \mathcal{R}^{d+1}$, where q_α indicates if feature α survive a corruption (the constant feature is never corrupted, hence $q_{d+1}=1$). If we denote the scatter matrix of the uncorrupted input as $S = \mathbf{X}\mathbf{X}^\top$, we obtain $E[\mathbf{R}]_{\alpha\beta} = \mathbf{S}_{\alpha\beta}\mathbf{q}_\alpha$ and

$$E[\mathbf{Q}]_{\alpha\beta} = \begin{cases} \mathbf{S}_{\alpha\beta}\mathbf{q}_\alpha\mathbf{q}_\beta & \text{if} \quad \alpha \neq \beta \\ \mathbf{S}_{\alpha\beta}\mathbf{q}_\alpha & \text{if} \quad \alpha = \beta \end{cases}. \tag{6}$$

The diagonal entries of $E[\mathbf{Q}]$ are the product of two identical features, and the probability of a feature surviving corruption is p. The expected value of the diagonal entries is therefore the scatter matrix multiplied by p. The off-diagonal entries are the product of two different features α and β, which are corrupted independently. The probability of both features surviving the corruption is p^2.

Squashing function. The output of the linear mapping $\mathbf{W} : \mathcal{R}^d \to \mathcal{R}^r$ approximates the expected value [17] of a prototype term. It can be beneficial to have more bag-of-word like features that are either present or not. For this purpose, we apply the tanh() squashing-function to the output

$$\mathbf{z} = \tanh(\mathbf{W}\mathbf{x}), \tag{7}$$

which has the effect of amplifying or dampening the feature values of the reconstructed prototype words. We refer to our feature learning algorithm as dCoT (Dense Cohort of Terms).

3.1 Recursive re-application

The linear mapping in eq.(7) is trained by reconstructing prototype words from partially corrupted input vectors. This linear approach works well for prototype words that commonly appear together with words of similar meaning (*e.g.* "president" and "obama"), as the mapping captures the correlation between the two. It can however be the case that two synonyms never appear together because the input documents are short and the authors use one term or the other but rarely both together (*e.g.* "tasty" and its rarer synonym "delicious"). In these cases it can help to recursively re-apply dCoT to its own output. Here, the first mapping reconstructs a common context between synonyms (*i.e.* words that co-occur with all synonyms) and subsequent applications of dCoT reconstruct the synonym-prototypes from this context. In the previous example, one could imagine that the first application of dCoT constructs context prototype words like "food", "expensive", "dinner", "wonderful" from the original term "delicious". The re-application of dCoT reconstructs "tasty" from these context words.

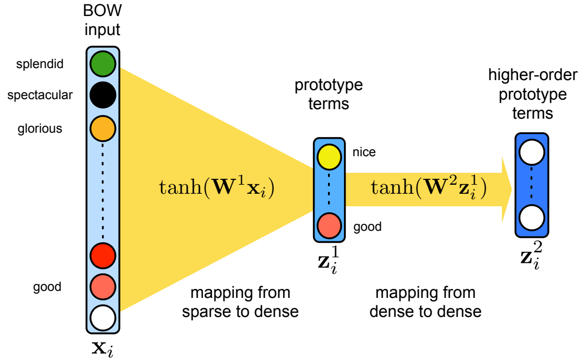

BOW input
splendid
spectacular
glorious
good
$\mathbf{x}_i$
$\tanh(\mathbf{W}^1\mathbf{x}_i)$
mapping from sparse to dense
prototype terms
nice
good
$\mathbf{z}_i^1$
$\tanh(\mathbf{W}^2\mathbf{z}_i^1)$
mapping from dense to dense
higher-order prototype terms
$\mathbf{z}_i^2$

Figure 1: Schematic layout of dCoT. The left part illustrates that dCoT learns a mapping from the overly sparse BoW representation to a dense one. The right part illustrates the recursive re-application to reconstruct prototype features from the context.

Let the mapping from eq.(7) be $\mathbf{W}^1 \in \mathcal{R}^{r\times d}$ and $\mathbf{z}_i^1 = \tanh\left(\mathbf{W}^1\mathbf{x}_i\right)$, for an input $\mathbf{x}_i$. We now compute a second mapping $\mathbf{W}^2 \in \mathcal{R}^{r\times r}$, exactly as defined in the previous section, except that we consider the vectors $\mathbf{z}_1^1, \dots, \mathbf{z}_n^1 \in \mathcal{R}^r$ as input. The mapping $\mathbf{W}^2$ is an affine transformation which stays within the prototype space spanned by P. This process can be repeated many times and because the input dimensionality is low the computation of (7) is cheap. Figure 1 illustrates this process in a schematic layout.

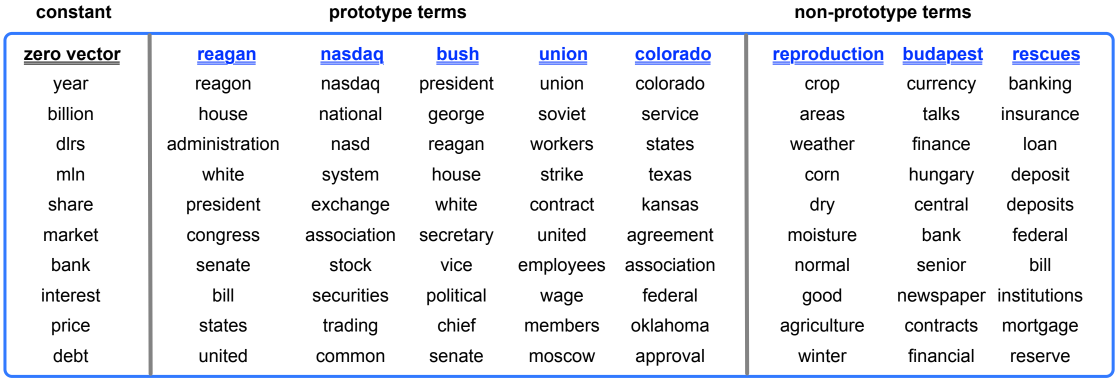
If dCoT is applied l times, the final representation $\mathbf{z}_i$ is the concatenated vector of all outputs and the original input,

$$\mathbf{z}_i = (\mathbf{x}_i, \mathbf{z}_i^1, \cdots, \mathbf{z}_i^l)^\top. \tag{8}$$

4. CONNECTION

dCoT shares some common elements with previously proposed feature learning algorithms. In this section, we discuss their similarities and differences.

Stacked Denoising Autoencoder (SDA). In the field of image recognition, the Autoencoder [18] and the Stacked Denoising Autoencoder (SDA) [19] are widely used to learn better feature representation from raw pixels input. dCoT shares several core similarities with SDA, which in fact inspired its original development. Similar to dCoT, SDA first corrupts the raw input, and learns to re-construct it. SDA also stacks several layers together by feeding the output of previous layers as input into sub-sequent layers. However,



constant	prototype terms					non-prototype terms		
zero vector	reagan	nasdaq	bush	union	colorado	reproduction	budapest	rescues
year	reagon	nasdaq	president	union	colorado	crop	currency	banking
billion	house	national	george	soviet	service	areas	talks	insurance
dlrs	administration	nasd	reagan	workers	states	weather	finance	loan
mln	white	system	house	strike	texas	corn	hungary	deposit
share	president	exchange	white	contract	kansas	dry	central	deposits
market	congress	association	secretary	united	agreement	moisture	bank	federal
bank	senate	stock	vice	employees	association	normal	senior	bill
interest	bill	securities	political	wage	federal	good	newspaper	institutions
price	states	trading	chief	members	oklahoma	agriculture	contracts	mortgage
debt	united	common	senate	moscow	approval	winter	financial	reserve

Figure 2: Term reconstruction from the Reuters dataset. Each column shows a different input term (*e.g.* "reagan", "nasdaq"), along with the prototype terms reconstructed from this particular input in decreasing order of feature values (top to bottom). The very left column shows the prototype terms generated by an all-empty input document.

the two algorithms also have substantial differences. The mapping in dCoT is a linear mapping from input to output (with a sub-sequent application of tanh()), which is solved in closed form. In contrast, SDA employs non-linear mapping from the input to a hidden layer and then to the output. Instead of a closed-form solution, it requires extensive gradient-descent-type hill-climbing. Further, SDA actually corrupts the input and is trained with multiple epochs over the dataset, whereas dCoT marginalizes out the corruption. In terms of running time, dCoT is orders of magnitudes faster than SDA and scales to much higher dimensional inputs [20, 21].

Principle Component Analysis (PCA). Similar to dCoT, Principle Component Analysis (PCA) [22] learns a lower dimensional linear space by minimizing the reconstruction error of the original input. For text documents, PCA is widely known through its variant as latent semantic indexing (LSI) [5]. Although both dCoT and LSI minimize reconstruction errors, the exact optimization is quite different. dCoT explicitly reconstructs prototype words from corruption, whereas LSI minimizes the reconstruction error after dimensionality reduction.

5. RESULTS

We evaluate our algorithm on *Reuters* and *Dmoz* datasets together with several other algorithms for feature learning.

Datasets. The *Reuters-21578* dataset is a collection of documents that appeared on Reuters newswire in 1987. We follow the convention of [23], which removes documents with multiple category labels. The dataset contains 65 categories, and consists of 5946 training and 2347 testing documents. Each document is represented by sBoW representation with 18933 distinct terms. The *Dmoz* dataset is a hierarchical collection of webpage links. The top level of the hierarchy consists of 16 categories. Following the convention of [24], we labeled each input by its top-level category, and remove some low-frequent terms. As a result, the dataset contains 7184 and 1796 training and testing points respectively, and each input is represented by the sBoW representation that contains 16498 distinct terms.

Reconstruction. Figure 2 shows example input terms (essentially one-word documents) and the prototype words that are reconstructed with dCoT on the Reuters dataset.

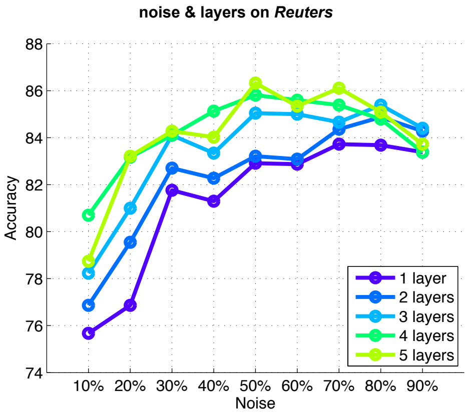


Figure 3: Classification accuracy trend on Reuters dataset with different layers and noise levels.

Each column represents a different input term (*e.g.* a document consisting of only the term "nasdaq") and shows the reconstructed prototype terms in decreasing order of their feature values (top to bottom). The very left column shows the prototype features generated by an all-empty input document. These features are completely determined by the constant bias, and coincide with the most frequent prototype terms in the whole corpora. For all other columns, we subtract this bias-generated vector to highlight the prototype words generated by the actual word and not the bias. As shown in the figure, two trends can be observed. First, prototype terms are reconstructed by other less common and more specific terms. For example, *president* is reconstructed by *reagan* and *bush*, and *stock* is reconstructed by *nasdaq*. Both *reagan* and *bush* are specific terms describing *president*. This trend indicates that dCoT learns the mapping from rare terms to common terms. Second, context and topics are reconstructed from rarer terms through the recursive re-application. For example, *agriculture* is reconstructed by *reproduction*, indicating that documents containing *reproduction* typically discuss topics related to *agriculture*. This connection indicates that dCoT also learns the higher order correlations between terms and topics.

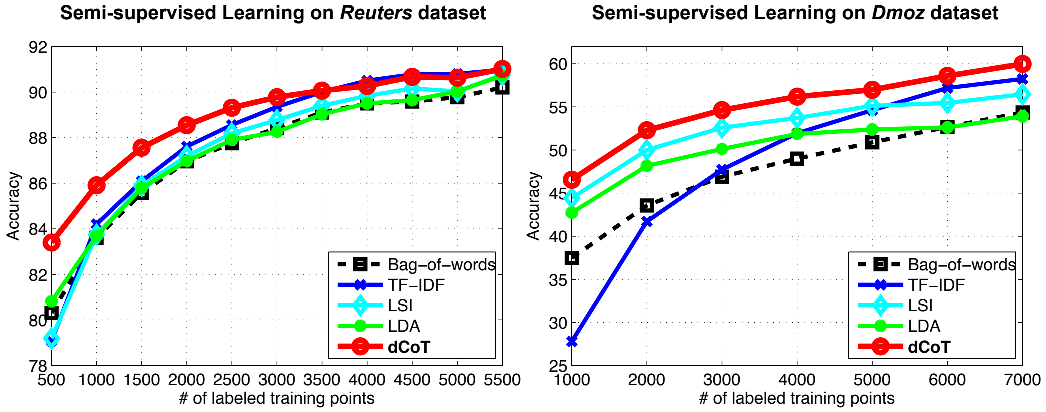


Figure 4: Semi-supervised learning results on the Reuters (left) and Dmoz (right) datasets. On both datasets, dCoT out-performs all other algorithms, especially when the number of labeled inputs is relatively small.

Parameter sensitivity. We also evaluate the effect of different noise level and number of layers (*i.e.* the number of recursive re-applications). Figure 3 shows the classification results on Reuters dataset as a function of layers l and noise level $1-p$. After training of dCoT (on the whole dataset), we randomly select $1,000$ labeled training inputs, train an SVM classifier [17] on the new feature representation, and test on the full testing set. Two trends emerge: 1. deep layers $l > 1$ improve over a single layered transformation — supporting our hypothesis that as we recursively re-apply dCoT, not only the feature representation is enriched, but also the higher order correlations between terms and topics are learned. 2. best results are obtained with a surprisingly high level of noise. We explain this trend by the fact that more corruption helps discover more subtle relationships between features and as we operate in the limit, and integrate out all possible corruptions, we can still learn even from substantially shortened documents.

Semi-supervised Experiments. In many real-world applications, the labeled training inputs are limited, because labeling usually involves human interaction and is expensive and time-consuming. However, unlabeled data is usually large and available. In this experiment we evaluate the suitability of dCoT to take advantage of semi-supervised learning settings. We learn the new feature representation with dCoT on the full training set (without labels), but train a linear SVM classifier on a small subset of labeled examples. We gradually increase the size of the labeled subset and evaluate on the whole testing set. For any given number of labeled training inputs, we average over five runs (of randomly picked labeled examples). We use the validation set to select the best combination of noise level and the number of layers.

As baselines, we compare against several alternative feature representations, which are all obtained from the full training set, similarly applied to a linear SVM classifier. The most basic baselines are the sBoW representation (with term frequency counts) and TF-IDF [1]. We compute the TF for each document separately, and obtain the IDF from the whole training set (including labeled and unlabeled data). We then apply the same IDF to the testing set. We also compare against latent semantic indexing (LSI) [5], for which we further split the training set into training and validation. We use the validation set to find the best parameter (numbers of leading Eigenvectors), and retrain on the whole training set with the best parameter. The new representation is obtained by projecting the sBoW feature space onto the LSI eigenvectors. Finally, we also compare against Latent Dirichlet Allocation (LDA) [4]. Similar to LSI, we use a validation set to find the best parameters, which include the Dirichlet hyper-parameter and the number of topics. The new representation learned from LDA are the topic mixture probabilities.

Datasets	TF-IDF	LSI	LDA	dCoT
Reuters	1s	51m	3h10m	2m
Dmoz	1s	1h38m	9h1m	3m

Table 1: Running time required for unsupervised feature learning with different algorithms.

The classification results are presented in figure 4. The graph shows that on both Dmoz and Reuters datasets, dCoT generally out-performs all other algorithms. This trend is particularly prominent in settings with relatively little labeled training data.

Running time. Table 1 compares the running times for feature learning with different algorithms. All timings are performed on a desktop with dual Intel^{TM}Six Core Xeon X5650 2.66GHz processors. Compared to LDA and LSI, the timing results show a three orders of magnitude speed-up on two datasets, reducing the feature learning time from several hours to a few minutes.

6. CONCLUSION

In this paper we present dCoT, an algorithm that efficiently learns a better feature representation for sBoW document data. Specifically, dCoT learns a mapping from high dimensional sparse to low dimensional dense representations by translating rare to common terms. Recursive re-application of dCoT on its own output results in the discovery of higher order topics from raw terms. On two standard benchmark document classification datasets we demonstrate that our algorithm achieves state-of-the-art results with very high reliability in semi-supervised settings.

Acknowledgment

This material is based upon work supported by the National Science Foundation under Grant No. 1149882. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

7. REFERENCES

[1] K. Jones, "A statistical interpretation of term specificity and its application in retrieval," *Journal of documentation*, vol. 28, no. 1, pp. 11–21, 1972.

[2] K. Weinberger, A. Dasgupta, J. Langford, A. Smola, and J. Attenberg, "Feature hashing for large scale multitask learning," in *Proceedings of the 26th Annual International Conference on Machine Learning.* ACM, 2009, pp. 1113–1120.

[3] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray, "Visual categorization with bags of keypoints," in *Workshop on statistical learning in computer vision, ECCV*, vol. 1, 2004, p. 22.

[4] D. Blei, A. Ng, and M. Jordan, "Latent dirichlet allocation," *The Journal of Machine Learning Research*, vol. 3, pp. 993–1022, 2003.

[5] S. Deerwester, S. Dumais, G. Furnas, T. Landauer, and R. Harshman, "Indexing by latent semantic analysis," *Journal of the American society for information science*, vol. 41, no. 6, pp. 391–407, 1990.

[6] H. Luhn, "A statistical approach to mechanized encoding and searching of literary information," *IBM Journal of research and development*, vol. 1, no. 4, pp. 309–317, 1957.

[7] G. Salton, A. Wong, and C. Yang, "A vector space model for automatic indexing," *Communications of the ACM*, vol. 18, no. 11, pp. 613–620, 1975.

[8] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Information processing & management*, vol. 24, no. 5, pp. 513–523, 1988.

[9] A. Singhal, C. Buckley, and M. Mitra, "Pivoted document length normalization," in *Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval.* ACM, 1996, pp. 21–29.

[10] K. Weinberger and O. Chapelle, "Large margin taxonomy embedding for document categorization," in *Advances in Neural Information Processing Systems 21*, D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, Eds., 2009, pp. 1737–1744.

[11] T. Hofmann, "Probabilistic latent semantic indexing," in *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval.* ACM, 1999, pp. 50–57.

[12] K. Nigam, A. McCallum, S. Thrun, and T. Mitchell, "Text classification from labeled and unlabeled documents using em," in *Machine Learning*, 1999, pp. 103–134.

[13] B. Bai, J. Weston, D. Grangier, R. Collobert, K. Sadamasa, Y. Qi, O. Chapelle, and K. Weinberger, "Supervised semantic indexing," in *Proceeding of the 18th ACM conference on Information and knowledge management*, ser. CIKM '09. New York, NY, USA: ACM, 2009, pp. 187–196.

[14] G. Golub and C. Reinsch, "Singular value decomposition and least squares solutions," *Numerische Mathematik*, vol. 14, no. 5, pp. 403–420, 1970.

[15] J. Blitzer, K. Weinberger, L. K. Saul, and F. C. N. Pereira, "Hierarchical distributed representations for statistical language modeling," in *Advances in Neural and Information Processing Systems*, vol. 17. Cambridge, MA: MIT Press, 2005.

[16] K. Weinberger, F. Sha, and L. K. Saul, "Learning a kernel matrix for nonlinear dimensionality reduction," in *Proceedings of the Twenty First International Conference on Machine Learning (ICML-04)*, Banff, Canada, 2004, pp. 839–846.

[17] C. Bishop, *Pattern Recognition and Machine Learning.* Springer, 2006.

[18] G. Hinton and R. Zemel, "Autoencoders, minimum description length, and helmholtz free energy," *Advances in neural information processing systems*, pp. 3–3, 1994.

[19] P. Vincent, H. Larochelle, Y. Bengio, and P. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proceedings of the 25th international conference on Machine learning.* ACM, 2008, pp. 1096–1103.

[20] M. Chen, Z. Xu, K. Q. Weinberger, and F. Sha, "Marginalized denoising autoencoders for domain adaptation," in *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, ser. ICML '12, J. Langford and J. Pineau, Eds. New York, NY, USA: ACM, July 2012, pp. 767–774.

[21] Z. E. Xu, K. Q. Weinberger, and F. Sha, "Rapid feature learning with stacked linear denoisers," *CoRR*, vol. abs/1105.0972, 2011.

[22] I. Jolliffe and MyiLibrary, *Principal component analysis.* Wiley Online Library, 2002, vol. 2.

[23] D. Cai, X. Wang, and X. He, "Probabilistic dyadic data analysis with local and global consistency," in *Proceedings of the 26th Annual International Conference on Machine Learning (ICML 2009)*, 2009, pp. 105–112.

[24] C. Do and A. Ng, "Transfer learning for text classification," in *Advances in Neural Information Processing Systems 18*, Y. Weiss, B. Schölkopf, and J. Platt, Eds. Cambridge, MA: MIT Press, 2006, pp. 299–306.