論文 / 著書情報
Article / Book Information

Title	Vocabulary Expansion Using Word Vectors for Video Semantic Indexing
Author	Nakamasa Inoue, Koichi Shinoda
Citation	Proc. ACM Multimedia, , , pp. 851-854
Issue date	2015, 10
Copyright	Copyright (c) 2015 Association for Computing Machinery
Note	This is the author's version of the work Proc. ACM Multimedia,
Set statement	(c) ACM, 2015. This is the author's version of the work. It is posted here by permission of ACM for your personal use. Not for redistribution. The definitive version was published in Proceedings of ACM Multimedia 15, 10. 26, 2015, 10. 30, 2015, http://dx.doi.org/10.1145/2733373.2806347

Vocabulary Expansion Using Word Vectors for Video Semantic Indexing *

Nakamasa Inoue
Tokyo Institute of Technology, Tokyo, Japan
inoue@ks.cs.titech.ac.jp

Koichi Shinoda
Tokyo Institute of Technology, Tokyo, Japan
shinoda@cs.titech.ac.jp

ABSTRACT

We propose vocabulary expansion for video semantic indexing. From many semantic concept detectors obtained by using training data, we make detectors for concepts not included in training data. First, we introduce Mikolov's word vectors to represent a word by a low-dimensional vector. Second, we represent a new concept by a weighted sum of concepts in training data in the word vector space. Finally, we use the same weighting coefficients for combining detectors to make a new detector. In our experiments, we evaluate our methods on the TRECVID Video Semantic Indexing (SIN) Task. We train our models with Google News text documents and ImageNET images to generate new semantic detectors for SIN task. We show that our method performs as well as SVMs trained with 100 TRECVID example videos.

Categories and Subject Descriptors

I.4 [**Computing Methodologies**]: Image Processing and Computer Vision; I.4.9 [**Applications**]: Multimedia Search and Recommendation

Keywords

Semantic Indexing; Video Search; Word Vectors; Deep Learning

1. INTRODUCTION

A large amount of video data has made been available on the Internet with advances in information and communication technologies. Semantic indexing is one of the most important techniques in multimedia applications including video search, surveillance and robot vision. Here, semantic indexing is the process to detect semantic concepts of objects, events, and scenes, e.g., *Bus*, *Airplane*, *Dancing*, and *Landscape*. Since the diversity of semantic concepts is very large, it has been a challenging task to bridge the semantic gap between low-level features and high-level semantics.

Most previous studies have focused on supervised learning methods such as support vector machines (SVMs) with Fisher vectors [1], and deep convolutional neural networks (CNNs) [2]. The Fisher vectors extend bag-of-visual-words [10] to a probabilistic framework by introducing Gaussian mixture models. SVMs are often trained for each semantic concept with them. Deep CNNs [2] typically have more than five layers for convolution and pooling, in which the input is a raw image and the output is a set of detection scores for each semantic concept. With most of these approaches, training images are required to be manually annotated. For example, to train a CNN for detecting object categories, several thousands of training images or videos are needed for each category.

Several methods use text data appeared with each image/video to decrease such annotation costs. For example, Joint embedding [3, 4] estimates a concept category from the text data appearing with their images at the same time, for example, on the same web page. But such a co-occurrence based approach cannot be used to detect an unseen concept in training data, i.e., a concept whose corresponding text data are not available in the training phase.

In natural language processing, word embedding methods have shown to be effective to capture semantic relationships between words. Their typical applications include machine translation [5], and web-document classification [6]. For example, word-vector representation proposed by Mikolov et. al [7] represents a word by a low-dimensional vector. They reported that the distance between semantically similar words tends to be smaller in the word vector space. Its interesting property is that semantic regularities are captured with vector operations for addition and subtraction. For example, a result of $\phi(\text{King}) - \phi(\text{Man}) + \phi(\text{Woman})$ is close to $\phi(\text{Queen})$ where $\phi(w)$ is a word vector of a word w. This may be also helpful to explore the relationship between semantic concepts in video.

In this paper, we propose a vocabulary expansion method based on word vectors for semantic indexing. From pretrained semantic-concept detectors, we make detectors for out-of-vocabulary concepts, i.e., concepts not included in training data, by using word vectors. To the best of our knowledge, applying word vectors to vocabulary expansion for video semantic indexing is novel. By introducing word vectors, we expect that our framework provides general semantic concept detectors, which are effective for detecting various semantic concepts in Internet videos. In our exper-

*Author's final version

 DOI: http://dx.doi.org/10.1145/2733373.2806347

iments, our method is evaluated on the TRECVID Video Semantic Indexing Task [9]. We train our model without TRECVID data by using the Google News documents [7] and the ImageNET [8] dataset and show that our method performs as well as SVMs trained with 100 example videos for each TRECVID semantic concept with Fisher-vector representation.

The rest of this paper is organized as follows. Section 2 presents the proposed framework. Section 3 shows experimental evaluations, and Section 4 describes conclusion and future work.

2. PROPOSED METHOD

Let x be a video and $\mathcal{W}$ be a set of words. We assume the word vectors $\phi(w) \in \mathbb{R}^d$ for all the words in $\mathcal{W}$ are obtained by introducing a word embedding method (e.g.,[7]). Our goal is to build visual concept detectors for each word.

The easiest way is to train detectors $f_w(x)$ (e.g., SVMs) for each $w \in \mathcal{W}$. However, this is not always a reasonable solution for real applications such as video search allowing users to enter any query words since the size of $\mathcal{W}$ is often large. To solve this problem, we build a detector for $w \notin \mathcal{C}$ from pre-trained detectors by using word vectors, where we assume pre-trained detectors for a subset $\mathcal{C} \subset \mathcal{W}$ are given.

We first approximate a word vector for $w \notin \mathcal{C}$ with the word vectors for words in $\mathcal{C}$ by

$$\bar{\phi}(w) = \sum_{i=1}^{N} \alpha_i \phi(w_i), \tag{1}$$

where $\bar{\phi}(w)$ is an approximation of a word vector $\phi(w)$, N is the number of words in $\mathcal{C}$, w_i is the i-th word in $\mathcal{C}$, and α_i is a weighting coefficient. We then use the same weighting coefficients for making a detector by

$$f_w(x) = \sum_{i=1}^{N} \alpha_i f_{w_i}(x). \tag{2}$$

Here we assume that those concepts (words) semantically similar to each other have similar visual features.

The following subsections present three methods using 1) vector quantization (VQ), 2) k-nearest neighbor search (KNN), and 3) Gaussian mixture models (GMMs), to determine the weighting coefficients in Eq. (1).

2.1 Vector Quantization

Approximating a word by the most similar word by applying vector quantization to word vectors is the simplest way to obtain the weighting coefficients. α_i is given by

$$\alpha_i = \begin{cases} 1 & \text{if } i = \underset{i}{\operatorname{argmin}}\, d(\phi(w), \phi(w_i)) \\ 0 & \text{otherwise} \end{cases}, \tag{3}$$

where $d(\cdot, \cdot)$ is the distance between two word vectors. For example, cosine distance between word vectors [7] can be introduced to here.

2.2 k-Nearest Neighbor Search

The k-nearest neighbor (KNN) search, which provides k similar words, is an extension of VQ to assign more than one similar words to a given word. With KNN, we set the weighting coefficients as

$$\alpha_i = \begin{cases} \frac{1}{k} & \text{if } i \in \mathrm{KNN}(w) \\ 0 & \text{otherwise} \end{cases}, \tag{4}$$

where $\mathrm{KNN}(w)$ is a set of k nearest words of w based on the distance measure $d(\cdot, \cdot)$ in Sec. 2.1.

2.3 Gaussian Mixture Models

Gaussian mixture models (GMMs) capture variance information for each pre-defined words and provide soft weighting for α_i as

$$\alpha_i = \frac{\pi_i \mathcal{N}(\phi(w)|\phi(w_i), \Sigma_i)}{\sum_{i=1}^{N} \pi_i \mathcal{N}(\phi(w)|\phi(w_i), \Sigma_i)}, \tag{5}$$

where $\phi(w)$ is a word vector, π_i is a mixture coefficient, and $\mathcal{N}(\cdot|\phi(w_i), \Sigma_i)$ is a pdf of Gaussian distribution with a mean vector $\phi(w_i)$ and a covariance matrix Σ_i for the i-th mixture component. Note that the word vectors for pre-determined words are used as mean vectors.

To estimate the covariance matrixes and the mixture coefficients from the set of word vectors for $\mathcal{W}$, we use the Expectation Maximization (EM) algorithm. Note that $\mathcal{W}$ includes words $w \notin \mathcal{C}$. In the M-step, the parameters are updated by

$$\pi_i = \frac{1}{|\mathcal{W}|} \sum_{w \in \mathcal{W}} c_{iw}, \tag{6}$$

$$\Sigma_i = \frac{1}{\sum_{w \in \mathcal{W}} c_{iw}} \sum_{w \in \mathcal{W}} c_{iw} (\phi(w) - \phi(w_i))(\phi(w) - \phi(w_i))^T \tag{7}$$

where c_{iw} is a *responsibility* of w to the i-th mixture component. The responsibilities are computed in the E-step as

$$c_{iw} = \frac{\pi_i \mathcal{N}(\phi(w)|\phi(w_i), \Sigma_i)}{\sum_{i=1}^{N} \pi_i \mathcal{N}(\phi(w)|\phi(w_i), \Sigma_i)}. \tag{8}$$

For the initialization values, we use $\pi_i = \frac{1}{N}$ and $\Sigma_i = I$ where I is the identity matrix.

2.4 Extension to a Set of Words

To further extend the discriminative function f to accept a set of words e.g., Airplane+Flying, we simply make the average vector as follows. Let $s = \{w^t\}_{t=1}^{T}$ be a set of word vectors. The average vector $\phi(s)$ is given by

$$\phi(s) = \sum_{t=1}^{T} \phi(w^t). \tag{9}$$

By approximating this vector in the same way as Eq. (1), the three methods presented in Sec. 2.1 to 2.3 are applied in the same way to build a detector for s.

3. EXPERIMENTS

3.1 Experimental Setup

We evaluate our method on the TRECVID 2010 Semantic Indexing dataset, which consists of 11,556 Internet video clips with creative commons licenses. Each video clip is divided into video shots based on the switching of a camera. Shot boundaries and a key-frame for each shot are provided with the dataset. The total number of video shots is 264,673, which is divided into 119,685 for training and 144,988 for testing. Only the key-frame images of the testing set are used to evaluate our methods.

The task is to detect 346 semantic concepts including objects, events, and scenes, e.g., Boat, AirplaneFlying, Dancing, Walking, Basketball and Cityscape, from video shots.

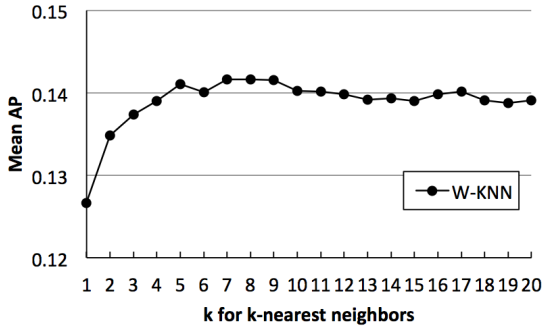

Figure 1: Evaluation using different k for k-nearest neighbor search.

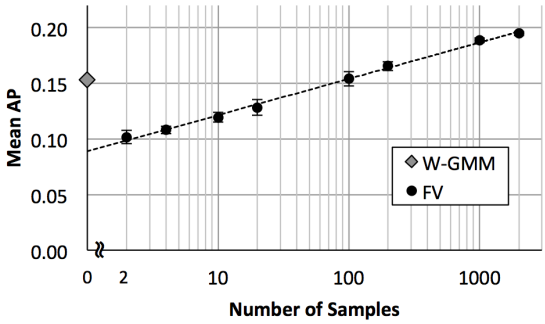

Figure 2: Comparison with Fisher vectors using different numbers of training samples.

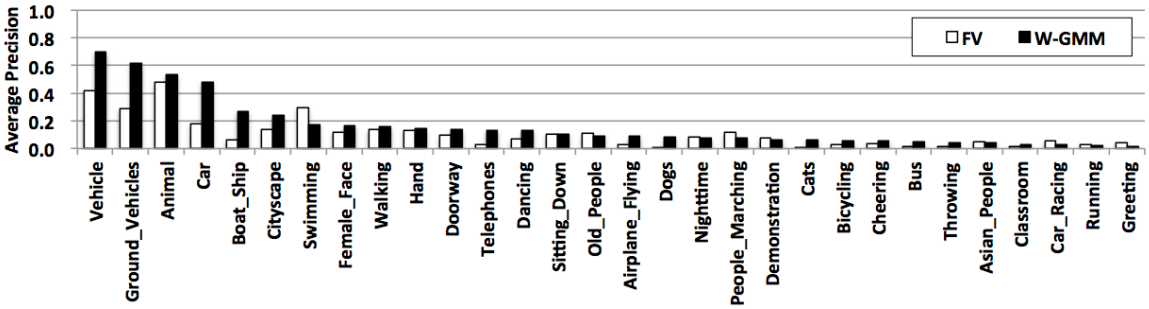

Figure 3: Average Precision (AP) by semantic concepts. AP for randomly sampled 30 of 346 semantic concepts are reported. FV: Fisher vectors of dense SIFT features. SVMs are trained on randomly sampled two examples, one for positive and one for negative. W-GMM: our word-based model using a GMM (Sec 2.3).

Table 1: Mean AP on TRECVID by methods. W-VQ, W-KNN, W-GMM: our word-vector based methods presented in Sec. 2, Fisher Vector: SVMs trained on one positive and one negative samples with Fisher vectors.

Method	Mean AP
W-VQ	0.127
W-KNN ($k = 10$)	0.140
W-GMM	**0.153**
Fisher Vector (one example) [1]	0.101

For semantic concepts presented by more than one word, we apply the method in Sec. 2.4. The evaluation measure is Mean Average Precision (Mean AP) over the 346 semantic concepts [1].

For the pre-trained concept detectors in Sec. 2, a CNN trained with 1,000 objects in the ImageNET Challenge 2012 [8] is used with the Caffe implementation [15]. Note that semantic concepts on TRECVID and the objects on ImageNET are exclusive. For the distance measure between words, cosine distance between 300-dimension word representations in [7] is used. This measure is trained on the Google News dataset, which has 100 billion words with 3 million synsets. We use the word2vec implementation [7]. We compare our methods with a baseline using SVMs with Fisher vectors (FV) trained for each 346 semantic concepts trained by using annotations provided in the TRECVID dataset. We follow the FV extraction process with dense SIFT features in [1].

[1]We use annotations provided in TRECVID 2014 to compute average precision, since the official annotations in 2010 only has 30 semantic concepts.

3.2 Experimental Results

Main Results

Table 1 compares Mean AP for word-based vector quantization (W-VQ), k-nearest neighbor search (W-KNN), and the Gaussian mixture model (W-GMM). We see that W-GMM outperforms the others. This shows that soft weighting in a GMM is effective for representing a semantic concepts by pre-determined objects.

For the parameter k in W-KNN, $k \geq 5$ is reasonable as shown in Figure 1. However, W-GMM is better than the best result of W-KNN using different k. This shows the effectiveness of probabilistic estimation of word distribution presented in Sec. 2.3.

Comparison with Fisher Vector

Comparison with Fisher Vector (FV) representation [1] is shown in Table 1. With FV, SVMs are trained for each

semantic concept in TRECVID with randomly sampled two examples, one for positive and one for negative. The Mean AP shown in the table is the average value of ten repeated experiments. We see that the FV performs worse than our methods. This is because the number of training samples is not enough to train SVMs.

In Figure 2, the number of training samples for FV is increased from 2 to 2,000. The result of W-GMM is also plotted on the figure. We conclude that W-GMM, which uses no examples from TRECVID, performs as well as SVMs trained with 100 example videos for each TRECVID semantic concept.

Analysis

Figure 3 shows AP for randomly sampled 30 of 346 semantic concepts. We see that our W-GMM performs well for objects such as vehicles and animals. As shown in Figure 4, which analyzes the relation between word vectors and visual features, vehicles and animals are separated into two clusters. This confirms our assumption that semantically similar concepts tend to have similar visual features. We conclude that this tendency helped to detect concepts in TRECVID that are not included in training data.

On the other hand, it is difficult to detect actions from video. For example, for a concept of "Swimming", concepts about actions with motion detectors and/or motion features should be added to the pre-trained detector to improve the action detection performance.

4. CONCLUSION

We proposed vocabulary expansion for video semantic indexing without any additional training with annoations. From pre-trained semantic-concept detectors, we made detectors for out-of-vocabulary concepts. Our experiments showed that the word-vector based Gaussian mixture models (W-GMM) performed the best and as well as SVMs trained with 100 example videos for each TRECVID semantic concept. Our future work will focus on motion analysis for semantic indexing to detect actions and events in videos.

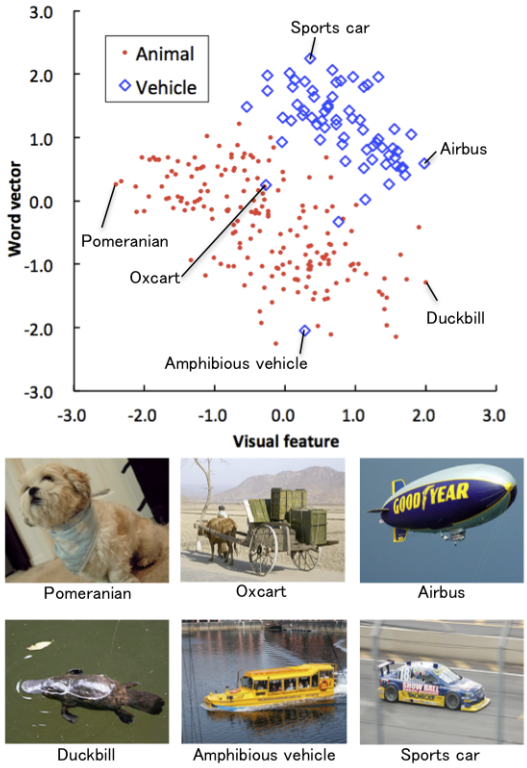

Figure 4: Analysis for Vehicles and Animals. The first principal components for word vectors and visual features are used for visualization. Visual features are obtained from the penultimate layer of the CNN, and are averaged on randomly sampled 100 images for each concept.

5. REFERENCES

[1] F. Perronnin, S. Jorge, and T. Mensink. Improving the fisher kernel for large-scale image classification. *Proc. ECCV*, pp.143–156, 2010.

[2] A. Krizhevsky, I.Sutskever, and G.E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. *Proc. NIPS*, pp.1–9, 2012.

[3] J. Weston, et al. Large Scale Image Annotation: Learning to Rank with Joint Word-Image Embeddings. *Porc. ECML*, 2010

[4] A. Frome, et al. Devise: A deep visual-semantic embedding model. *Proc. NIPS*, 2013.

[5] T. Mikolov, et al. Distributed Representations of Words and Phrases and their Compositionality. *Proc. NIPS*, 2013.

[6] D. Jacob, et al. Fast and Robust Neural Network Joint Models for Statistical Machine Translation. *Proc. ACL*, 2014.

[7] T. Mikolov, et al. Efficient Estimation of Word Representations in Vector Space. *Proc. ICLR*, 2013.

[8] J. Den, et al. ImageNet: A Large-Scale Hierarchical Image Database. *Proc. CVPR*, 2009.

[9] P. Over, et. al. TRECVID 2013 – An Overview of the Goals, Tasks, Data, Evaluation Mechanisms and Metrics. *Proc. TRECVID workshop*, 2013.

[10] G. Csurka, et al. Visual categorization with bags of keypoints. *Proc. ECCV SLCV*, pp.59–74, 2004.

[11] K. Chatfield, et al. The devil is in the details: an evaluation of recent feature encoding methods. *Proc. BMVC*, pp.1–12, 2011.

[12] J. Wang, et al. Locality-constrained linear coding for image classification. *Proc. CVPR*, pp.3360–3367, 2010.

[13] X. Zhou, et al. Image classification using super-vector coding of local image descriptors. *Proc. ECCV*, pp.141–154, 2010.

[14] C.G.M. Snoek, et. al. The MediaMill TRECVID 2012 Semantic Video Search Engine. *Proc. TRECVID workshop*, 2012.

[15] Y. Jia, et al. Caffe: Convolutional Architecture for Fast Feature Embedding. *Proc. ACM Multimedia Open Source Competition*, 2014.