BAG OF WORDS MODEL FOR TEXTURE CLASSIFICATION

Jesper Hybel, Belgrano Lorenzo, Jensen Søren, Mirza Hasanbasic

Students at DTU, Denmark

ABSTRACT

In this article we present and illustrate a method of picture segmentation based on texture recognition and show that the texture recognition model is transferable from one dataset to another. The method for picture segmentation combines a region proposal algorithm with a texture recognition method based on the convolutional layers of VGG-16 in combination with fischer vectors. We test the texture recognition method on the uncluttered textures using Describable Textures Dataset (DTD) and transfer it on cluttered textures using the OpenSurfaces (OS) dataset allowing only partial retraining. We achieve a classification accuracy of 64.0% on the DTD dataset and 56.48% on the OS dataset.

Index Terms— Convolutional neural networks , Fisher Vectors, VGG-16, Texture Classification, Benchmark,

1. INTRODUCTION

In recent years, the use of computer vision has been popularized in research areas such as autonomous cars [1] and automatic image captioning [2]. Just as the vision is important to humans, when computer vision fails, the consequences can be detrimental [3, 4].

To achieve computer vision, two steps are often implemented. First, a region proposal algorithm is used to introduce the image regions which contain objects in the image. Hereafter, the object in each of the proposed regions is recognized using a classification algorithm.

Traditionally computer vision was achieved using manually designed feature extraction schemes such as Scale Invariant Feature Transform (SIFT). In [5], P. Viola and M. Jones presented an algorithm for detection of human faces. In this work, the manually designed features were first extracted from the image. Hereafter, a Support Vector Machine (SVM) was trained to separate the features describing human faces from the background. In 2005, N. Dalal and B. Triggs presented the Histogram of Oriented Gradients (HOG) [6]. Here, the light gradients of the image was extracted as a sparse representation of the image. Hereafter, an SVM was again implemented on a local image region for human detection.

The traditional methods of computer vision were susceptible to changes in the image, such as rotation of the classification object. In 2012, K. Alex et al. presented a neural network based method (Alex-Net). In this method, a series of convolutional layers learned to extract the image features, and a series of dense network layers decoded the features to the object proposal. Although convolutional layers had been known for a long time, the increase in computational power and larger datasets allowed the neural network to generalize well.

Many improvements of the neural network framework in Alex-Net has since been proposed to speed up the process. In [7], a region proposal method is introduced before the neural network, which allows for faster recognition of the objects. J. Redmon et al. proposes in [8, 9] to only feed the image once to the convolutional neural network. The network then simultaneously predicts several bounding boxes and corresponding classes.

In this article we propose a region segmentation scheme based on texture recognition. A region proposal method is first used to give several overlapping proposals for image regions. These regions are then propagated through a Convolutional Neural Network (CNN) and Fisher Vectors (FV) are hereafter computed from the features. Hereafter, an SVM is trained to classify the regions and the majority vote is used to estimate the texture of all overlapping regions. We hypothesize that a pretrained CNN can be used to extract the low level features through transfer learning. Furthermore, we believe that texture classification can lead to a robust method of region segmentation which is invariant to artifacts such as shadows.

The remainder of this article is organized as follows: Section 2 describes the proposed method and presents the data used for model evaluation. Results obtained by evaluating the method on the selected datasets are given in section 3. Hereafter, the results are discussed in section 4 before a conclusion and final remarks are given in section 5.

2. MATERIALS AND METHODS

2.1. The Data

We use the Describable Textures Dataset (DTD) which consist of $n = 1, ..., N = 5640$ texture images $\mathbf{z}_n$ balanced across 47 texture labels. Textures fill at least 90% of the image in order to allow studying the problem of texture description

independently of texture segmentation.

We also use a subset of the OpenSurfaces (OS) dataset which comprises $N = 10.422$ images. In these images the textures appear cluttered and so segmentation is necessary. However evaluation are restricted to 11 of the 47 texture types because only these occur with frequency higher than 100.

The images in both data sets are collected from the internet rather than experimentally generated. Arguably this makes the data more representative of the real-world variations met in applications [10].

2.2. The Model

The model used for texture classification can be described as a composition with three components

$$\mathbf{y}_c = \phi_C(\mathbf{z}) = (\phi_{SVM} \circ \phi_{FV} \circ \phi_{CNN})(\mathbf{z}), \tag{1}$$

where $\mathbf{z} \in \mathbb{R}^H \times \mathbb{R}^W \times \mathbb{R}^3$ is the input picture and $\mathbf{y}$ is the resulting texture label.

The successive steps of the model are as follows: First a convolutional neural network $\phi_{CNN}(.)$ is applied in order to extract local image descriptors. Secondly the image descriptors are use to compute the corresponding Fischer Vector using the orderless pooling encoder $\phi_{FV}(.)$. Lastly a classifier is applied, which in this case is a support vector machine $\phi_{SVM}(.)$.

When the texture classification is used for segmentation the successive steps are altered

$$\mathbf{y} = \phi(\mathbf{z}) = (\phi_{MV} \circ \phi_c \circ \phi_R)(\mathbf{z}), \tag{2}$$

to include a step of region proposals $\phi_R(.)$ as a first step and $\phi_{MV}(.)$ as a final step of majority voting in order to classify each pixel.

2.2.1. Convolutional Neural Networks

For the feature extraction we use the convolutional layers of a pre-trained *VGG-16* network. The network is implemented using Tensorflow with weights downloaded from [11]. We choose the VGG-16 network, instead of a more recent network because it strikes a good balance between efficiency and practicality with respect to implementation.

Each picture $\mathbf{z}_n$ is hereby transformed

$$\mathbf{x}_n = \phi_{CNN}(\mathbf{z}_n) \tag{3}$$

into $t = 1, ..., T$ *image descriptors* such that $\mathbf{x}_n = \{\mathbf{x}_{nt}\}_{t=1}^T$ with $\mathbf{x}_{nt} \in \mathbb{R}^{512}$, where the dimensionality is the number of feature channels of the CNN. The number of image descriptors T depend on the dimensionality $W \times H$ of the picture n and is approximately $W \cdot H/2^5$ due to the 5 max pooling layers in the CNN [11].

Truncating the CNN earlier, at the level of the convolutional layers should be seen as a way to get powerful local descriptors. The FC layers are order sensitive and therefore capture the spatial layout, which may not be needed for texture recognition. Instead we use the Fischer Vectors as introduced in the next section. Furthermore the deeper layers are likely to be more domain-specific and therefore potentially less transferable than the shallower convolutional layers. Lastly the input for the CNN, has to be fixed with FC layers and thus it can be expensive to re-size the input image.

2.2.2. Fisher Vectors

In this section we explain how we train a Gaussian Mixture Model (GMM) and use this in order to get from the image descriptors $\mathbf{x}_n$ for picture n to a Fischer Vector representation $\mathbf{s}_n$ of the image

$$\mathbf{s}_n = \phi_{FV}(\mathbf{x}_n). \tag{4}$$

Instead of pooling features using histograms, as in a traditional *Bag of Words* approach, we implement the fisher vectors, which will allow for a statistical description of the "visual words". We will model the extracted features as being generated by a *Gaussian Mixture Model*, in which each component acts as a "visual word". The generative distribution over the features is assumed given by

$$p(\mathbf{x}_{nt}|\boldsymbol{\theta}) \sim \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_{nt}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k), \quad \sum_{k=1}^{K} \pi_k = 1.$$

In our particular experimental framework, we used $K = 64$ mixture components to build the dictionary, assuming the covariance matrix of each component to be diagonal in order to avoid issues with singularities.

We can find the optimal set of parameters $\{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K$, which we also denote as $\boldsymbol{\theta}$, using *Maximum Likelihood* optimization, where we want to maximize the log-likelihood given by

$$\log \mathcal{L}(\boldsymbol{\theta}) = \sum_{n}^{N} \sum_{t}^{T} \log p(\mathbf{x}_{nt}|\boldsymbol{\theta}), \tag{5}$$

with a training set $\{\mathbf{x}_n\}_{n=1}^N$ given. This can be done by using the *Expectation-Maximization* algorithm [12].

Using the estimate $\hat{\boldsymbol{\theta}}$ from fitting the GMMs on a training set, the fisher vectors can now be calculated using the *normal-*

ized gradient for the likelihood

$$\mathbf{s}_n^{\boldsymbol{\pi}_k} = \frac{1}{\sqrt{\pi_k}} \sum_{t=1}^{T} (\gamma_k(\boldsymbol{x}_{nt}) - \pi_k) \tag{6}$$

$$\mathbf{s}_n^{\boldsymbol{\mu}_k} = \frac{1}{\sqrt{\pi_k}} \sum_{t=1}^{T} \gamma_k(\boldsymbol{x}_{nt}) \left(\frac{\boldsymbol{x}_{nt} - \boldsymbol{\mu}_k}{\boldsymbol{\sigma}_k} \right) \tag{7}$$

$$\mathbf{s}_n^{\boldsymbol{\sigma}_k} = \frac{1}{\sqrt{\pi_k}} \sum_{t=1}^{T} \gamma_k(\boldsymbol{x}_{nt}) \frac{1}{\sqrt{2}} \left[\left(\frac{\boldsymbol{x}_{nt} - \boldsymbol{\mu}_k}{\boldsymbol{\sigma}_k} \right)^2 - 1 \right]. \tag{8}$$

where the responsibilities are defined as

$$\gamma_k(\boldsymbol{x}_{nt}) = \frac{\pi_k \mathcal{N}_k(\boldsymbol{x}_{nt} | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}_j(\boldsymbol{x}_{nt} | \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}. \tag{9}$$

The resulting gradient scores are then stacked and normalized (using *power* and *L2* normalization) in order to get $\mathbf{s}_n$ of dimensionality $64 + 64 \cdot 512 + 64 \cdot 512$.

2.2.3. Classification: Support Vector Machine

For classification $\phi_{SVM}(.)$ we use linear SVM's fitted by minimizing the hinge loss. We use 1-vs-all SVM to classify textures according to the 47 classes in the DTD dataset and 11 classes on the OS dataset. For texture recognition on DTD, *Experiment 1*, we predict using the highest confidence score. For texture recognition on OS, *Experiment 2*, we do not predict a single class but instead evaluate each SVM on its own.

2.2.4. Region Proposals and Majority Voting

For classification on the OS dataset we use region proposals $\phi_R(.)$ due to the textures being cluttered. This dataset however comes with region proposal which we use in *Experiment 2*. For *image segmentation* we implement a *Multi-scale Combinatorial Grouping* method for region proposals[13]. This region proposal method provides segmentation masks which allows us to classify patches of the image individually. In this method, normalized cuts segmentation is performed on several downscaled and upscaled versions of the image. As upscaling by interpolation results in softer edges, and downscaling results in harder edges, the regions proposed will be different at each scale. Because the regions provided by the region proposal algorithm may contain several texture classes, we request a proposal scheme which provides several overlapping regions. Here we assume that each pixel will appear in more regions with its true texture than with other textures and can therefore do the pixel-by-pixel classification with majority voting $\phi_M(.)$.

3. RESULTS

In this section we present the results obtained by implementation of the proposed algorithm on the DTD dataset as well as a labeled subset of the OS dataset.

3.1. Texture Recognition

In *Experiment 1* we evaluate the texture recognition approach on the DTD dataset, using ten fold cross validation and then calculating the average accuracy. The performance of the approach can be seen in Table 1 along with comparative results found in the literature. We observe that changing the fully connected layer with fisher vectors increase performance by approximately 4 percentage points and note that the confidence intervals are non-overlapping. We also see that other proposed algorithms, using CNN's with more layers or residual networks, beat our results by a large margin. In the confusion matrix (Fig. 1) no systematic misclassification can be seen, even though some classes (for example class 2, "blotchy") are very poorly classified.

METHOD	DTD
SIFT + FV	$58.6_{\pm 1.2}$
VGG-16 + FC	$58.8_{\pm 0.8}$
FV-CNN (VGG-16)	$64_{\pm 1.0}$
FV+FC VGG-VD	$74.7_{\pm 1.0}$
TEX-Net-LF (ResNet)	$73.6_{\pm 0.6}$

Table 1: Classification accuracy and 95% confidence interval on the DTD dataset as obtained by different methods: SIFT and Fisher Vector pooling [14], VGG-16 with Fully Connected layers [10], VGG-16 with FVs (our method),*very deep* CNN w. FVs and FC [11] and Local Binary patterns encoded CNN models w. ResNet [15].

In the OS labeled dataset, we restrict our investigation to the eleven well represented classes. We first tried to classify the OS dataset with the classifiers obtained from the DTD dataset. However, this resulted in very poor performance. For what we choose to call *Experiment 2* we retrained the SVM's but not the GMM and performed 10-fold cross validation again. For the retrained classifiers, the most accurate class-predictor obtained an accuracy of 70.37% and the worst had accuracy of 38.23%. The average accuracy for all eleven classes was 56.48%. In table 2, we first note the very poor performance of the classification scheme without retraining. We also note that our method performs well when it is retrained, but is still more than 11%-points worse than the state of the art.

3.2. Image Segmentation

To illustrate how the image segmentation algorithm works we provide an example as seen in Figure 2. In image (a) we

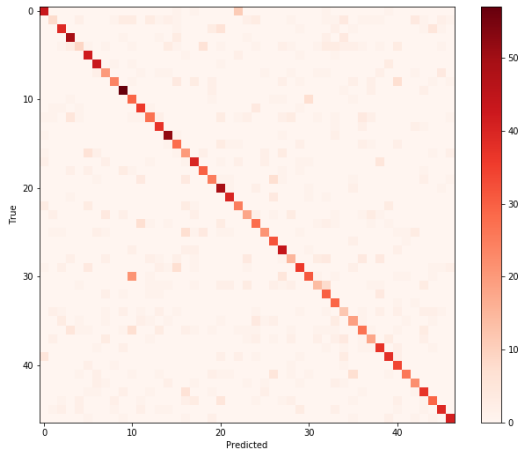

Fig. 1: Confusion matrix for the FV-CNN on the DTD texture classification class. The colorbar shows the class prediction accuracy.

METHOD	OS
SIFT + FV	39.8%
VGG-16 + FC	54.3%
FV-CNN (VGG-16) (from DTD)	22.65%
FV-CNN (VGG-16) (retrained)	56.48%
FV+FC VGG-VD	67.9%

Table 2: classification accuracy on the OS dataset in the proposed method as well as relevant litterature [10]

see and image which contains a "chequered" region. Image (c) shows us the region proposed by the image segmentation algorithm. This is clearly not the entire true region and the Intersection Over Union score from image (b) and (c) is 64.65%, which we believe is acceptable but far from perfect.

4. DISCUSSION

The results obtained on the DTD dataset clearly show that it is possible to do classification on the DTD dataset using our method. From this, it is also clear that transfer learning can be used for texture classification. We believe that the reason why this is possible is because the neurons in the VGG-16 network recognize features in the DTD dataset that it also learned to generalize during it's training. As an example, a "stripy" image may light up in the same neuron as an image of a zebra. We can also clearly see that replacing the fully connected layer with fisher vectors allowed for a 5% improvement in accuracy. Although we see improved results compared to SIFT-FV and VGG16-FC, we can clearly see that in the state of the art there exists even better methods.

Looking at the results from the OS dataset, it is clear that retraining of the SVM classifiers was needed in order to achieve a good performance. As the SVM classifiers try to

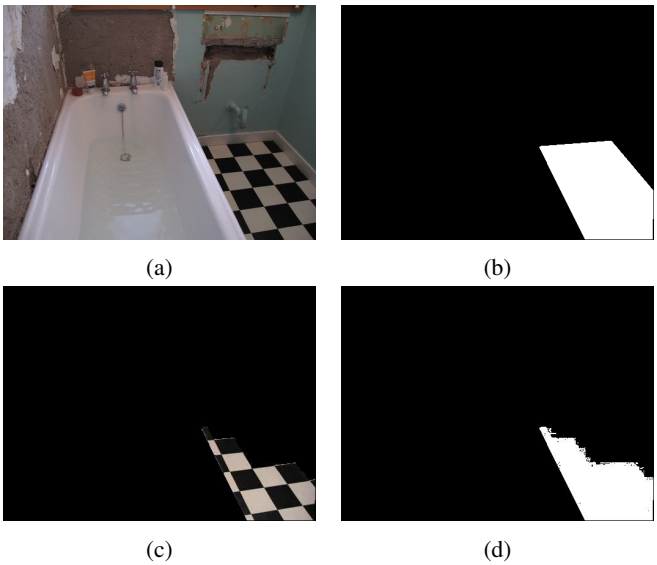

Fig. 2: Image segmentation using texture classification. a) original image with chequered region, b) true region mask, c) proposed region by image segmentation and d) mask of proposed region

seperate the fisher vectors of each class, it is clear that the fisher vectors for the OS classes are different from the fisher vectors for the corresponding DTD classes. Here we once again see that replacing a fully connected layer with fisher vectors increases the model's accuracy, and that our model again is beat by the state of the art.

For image segmentation, we see that it is possible to segment a a texture class from an image, although we didn't segment the entire true region. Here, the remaining chequered floor was classified as *zigzagged*, which is also what we would describe the missing border as. This means that even though the texture classes are different, some elements of each class could be overlapping which will disrupt the classifier.

5. CONCLUSION

In this article we presented a scheme for texture classification. We believed that it would be possible to use transfer learning by using a pre-trained CNN to extract low level features from the image. Through the article, we conducted experiments on the DTD and OS datasets, which confirmed this hypothesis. We also presented a way to use texture classification for image segmentation. Through an example we achieved an IOU score of 64.65%, however it was clear from the results that a section of the real segment was left out due to misclassification. Results from the DTD and OS datasets showed that replacing the fully connected layers with fisher vectors and SVM classification improved the classification accuracy.

6. REFERENCES

[1] Chenyi Chen, Ari Seff, Alain Kornhauser, and Jianxiong Xiao, "Deepdriving: Learning affordance for direct perception in autonomous driving," in *Computer Vision (ICCV), 2015 IEEE International Conference on*. IEEE, 2015, pp. 2722–2730.

[2] Andrej Karpathy and Li Fei-Fei, "Deep visual-semantic alignments for generating image descriptions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3128–3137.

[3] Ryan Calo, "Is the law ready for driverless cars?," *Communications of the ACM*, vol. 61, no. 5, pp. 34–36, 2018.

[4] Sunghyo Kim, "Crashed software: Assessing product liability for software defects in automated vehicles," *Duke Law & Technology Review*, vol. 16, no. 1, pp. 300–317, 2018.

[5] Paul Viola and Michael Jones, "Rapid object detection using a boosted cascade of simple features," in *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*. IEEE, 2001, vol. 1, pp. I–I.

[6] Navneet Dalal and Bill Triggs, "Histograms of oriented gradients for human detection," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*. IEEE, 2005, vol. 1, pp. 886–893.

[7] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik, "Region-based convolutional networks for accurate object detection and segmentation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 1, pp. 142–158, 2016.

[8] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[9] Joseph Redmon and Ali Farhadi, "Yolo9000: better, faster, stronger," *arXiv preprint*, 2017.

[10] M. Cimpoi, Kokkinos I. Maji, S., and A. Vedaldi, "Deep filter banks for texture recognition, description, and segmentation.," *International Journal of Computer Vision*, vol. 118 (6), pp. 65–94, 2016.

[11] Karen Simonyan and Andrew Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[12] M Bishop Christopher, *PATTERN RECOGNITION AND MACHINE LEARNING.*, Springer-Verlag New York, 2016, pp. 438-441.

[13] Jon Barron Ferran Marques Jitendra Malik Pablo Arbeláez, Jordi Pont-Tuset, "Multiscale combinatorial grouping for image segmentation and object proposal generation," *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[14] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, , and A. Vedaldi, "Describing textures in the wild," in *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[15] Joost van de Weijer Matthieu Molinier Jorma Laaksonen Rao Muhammad Anwer, Fahad Shahbaz Khan, "Binary patterns encoded convolutional neural networks for texture recognition and remote sensing scene classification," *ISPRS Journal of Photogrammetry and Remote Sensing*, 2014.