SERCAN SUNETCI

M.S. Thesis

2017

EVALUATION OF FEATURE SELECTION AND ENCODING METHODS FOR SUPERPIXEL IMAGE PARSING

SERCAN SÜNETCİ

IŞIK UNIVERSITY
2017

EVALUATION OF FEATURE SELECTION AND ENCODING METHODS FOR SUPERPIXEL IMAGE PARSING

SERCAN SÜNETCİ
B.S., Electrical & Electronic Engineering, IŞIK UNIVERSITY, 2014
Submitted to the Graduate School of Science and Engineering
in partial fulfillment of the requirements for the degree of
Master of Science
in
Electronics Engineering

IŞIK UNIVERSITY
2017

IŞIK UNIVERSITY
GRADUATE SCHOOL OF SCIENCE AND ENGINEERING

EVALUATION OF FEATURE SELECTION AND ENCODING METHODS FOR SUPERPIXEL IMAGE PARSING

SERCAN SÜNETCİ

APPROVED BY:

Prof. Dr. Hasan F. Ateş (Thesis Supervisor)	Işık University	_____
Prof. Dr. Uluğ Bayazıt	Istanbul Technical University	_____
Assoc. Prof. Taner Eskil	Işık University	_____

APPROVAL DATE: 14/12/2017

EVALUATION OF FEATURE SELECTION AND ENCODING METHODS FOR SUPERPIXEL IMAGE PARSING

Abstract

This thesis is about image parsing which is one of the important problems in computer vision. The goal of image parsing is segmentation of object and labeling of each object.

Recently, a popular way of image segmentation and classification is superpixels. Image is segmented into visually logical small regions by using superpixel algorithm and then, superpixels are parsed into different classes. Classification performance is significantly affected by the properties of superpixel algorithm and parametric settings. SuperParsing is one of the superpixel-based image parsing algorithm and provides a succesful nonparametric solution for image segmentation and classification problem without any need for classifier training. SuperParsing labels each superpixel based on feature matching between the superpixel and a subset of the training superpixels. The training subset is determined by global matching between the test image and the training set. For superpixel matching the method makes use of a rich set of superpixel features. Class conditional log-likelihood is computed based on these matched features.

The main objective of this thesis is to show improvements in labeling accuracy percentage by using feature encoding and selection methods, including learned features from Convolutional Neural Network (CNN) models. We perform two different encoding methods to selected features of superpixels and show that feature encoding improves parsing accuracy. The applied feature encoding methods are locality-constrained linear encoding (LLC) and kernel codebook encoding (KCB). LLC encoding method gives us 2.6% improvement on per-pixel accuracy for SIFT Flow dataset and 6.8% improvement on per-pixel accuracy for 19-class LabelMe dataset. KCB encoding method gives us 3.6% improvement on per-pixel accuracy for SIFT Flow dataset and 6.2% improvement on per-pixel accuracy for 19-class LabelMe dataset. All these results are overall improvement which are computed over original SuperParsing.

Most recent studies about image segmentation and classification use CNN to improve their accuracy percentage. Features extracted from pre-trained networks, which are trained on large image databases, can be used in addition to hand-crafted features in image segmentation. Last layer of these CNN models give the best features for classification. We test learned CNN features together with KCB or LLC encoding methods. We use CNN features both for global matching and superpixel matching. These tests give us 7.3% overall improvement over original SuperParsing on SIFT Flow dataset and 10.3% overall improvement over original SuperParsing on 19-class LabelMe dataset.

Keywords: image parsing, feature encoding, image segmentation, image classification, cnn models

SÜPERPİKSEL İMGE AYRIŞTIRMASI İÇİN ÖZNİTELİK SEÇİMİ VE KODLAMA YÖNTEMLERİNİN DEĞERLENDİRMESİ

Özet

Bu tez, bilgisayarla görünün önemli problemlerinden olan görüntü ayrştırma ile ilgilidir. Görüntü ayrıştırmanın amacı nesnenin bölütlenmesi ve her bir nesnenin etiketlenmesidir.

Son zamanlarda imge bölütleme ve sınıflandırmanın popüler yolu süperpiksellerdir. Görüntü, süperpiksel algoritması kullanlarak grsel olarak küçük mantıksal bölgelere bölünür. Daha sonra süperpikseller farklı sınıflara ayrılır. Sınıflandırma performansı süperpiksel algoritmasının özelliklerinden ve parametre ayarlardan önemli ölçüde etkilenmektedir. SuperParsing, süperpiksel tabanlı bir görüntü ayrıştırma algoritmasıdır. Bu algoritma herhangi bir sınıflandırıcıya ihtiyaç duymadan başarılı bir parametrik olmayan çözüm sağlar. SuperParsing her bir süperpikseli süperpiksel ve eğitim süperpiksellerinin altkümesi arasındaki öznitelik eşlemesine bağlı olarak etiketler. Bu eğitim altkümesi test görüntüsü ve eğitim kümesi arasındaki global eşleme tarafndan belirlenir. Bu yöntem süperpiksel eşleme için süperpiksel özelliklerinin zengin bir kümesini kullanır. Koşullu sınıf olabilirliği bu eşlenmiş özniteliklere bağlı olarak hesaplanır.

Bu tezin temel amacı Evrişimsel Sinir Ağı (ESA) modellerinden öğrenilmiş öznitelikleri içeren öznitelik kodlama ve seçim yöntemleri kullanılarak etiketleme doğruluğu yüzdesindeki gelişmeleri göstermektir. Süperpiksellerin seçilmiş özniteliklerine iki farklı kodlama yöntemi uyguluyoruz ve öznitelik kodlamanın ayrıştırma doğruluğunu geliştirdiğini gösteriyoruz. Yerellik-Kısıtlı Doğrusal (YDK) ve Kernel Kod-tablosu (KKT) gibi öznitelik kodlama yöntemleri uygulanmıştır. YDK kodlama yöntemi SIFT Flow veri kümesinde %2.6 ve 19 sınıflı LabelMe veri kümesinde ise %6.8 artış sağlamıştır. KKT kodlama yöntemi SIFT Flow veri kümesinde %3.6 ve 19 sınıflı LabelMe veri kümesinde ise %6.2 artış sağlamıştır. Tüm bu sonuçlar orijinal SuperParsing üzerinden hesaplanan toplam kazançtır.

Son zamanlardaki görüntü bölütleme ve sınıflandırma çalışmaların çoğunluğu doğruluk yüzdelerini geliştirmek için ESA kullanır. Görüntü bölütlemede büyük

görüntü veri tabanlarında eğitilmiş olan ön eğitimli ağlardan çıkartılan öznitelikler el yapımı özniteliklere ek olarak kullanılabilir. Bu ESA modellerinin son katmanları sınıflandırma için en iyi öznitelikleri verir. Öğrenilmiş ESA özniteliklerini KKT veya YDK kodlama yöntemleri ile birlikte test ettik. ESA özniteliklerini hem global eşleme hem de süperpiksel eşleme için kullandık. Bu testler orijinal SuperParsing üzerine SIFT Flow veri kümesinde %7.3 ve 19 sınıflı LabelMe veri kümesinde ise %10.3 toplam kazanç sağlamıştır.

Anahtar kelimeler: imge ayrıştırma, öznitelik kodlaması, imge bölütleme, imge sınıflandırması, cnn modelleri

Acknowledgements

The research was supported by The Scientific and Technological Research Council of Turkey (TÜBİTAK) Grant No: 115E307 and Işık Universtiy BAP project Grant No: 14A205.

First of all, I thank Prof. Dr. Hasan F. Ateş who is my project supervisor to always support me with his extensive knowledge in the all parts of the project.

Also, I thank to my parents Selçuk Sünetci, Berra Sünetci and my brother Berk Sünetci, who always support me with their love and patient.

Finally, I want to thank my friends, who always make me feel never alona with their friendship, patience and help during the academic years.

To my family. . .

Table of Contents

List of Tables

List of Figures

List of Abbreviations

2-D	**2** **D**imension
3-D	**3** **D**imension
SIFT	**S**cale-**I**nvariant **F**eature **T**ransform
LLC	**L**ocality-**C**onstrianed **L**inear Coding
KCB	**K**ernel **C**ode**b**ook Encoding
CNN	**C**onvolutional **N**eural **N**etwork
YDK	**Y**erellik-**K**ısıtlı **D**oğrusal Kodlama
KKT	**K**ernel **K**od-**t**ablosu Kodlama
ESA	**E**vrişimsel **S**inir **A**ğı Modeli
MRF	**M**arkov **R**andom **F**ield
MRA	**M**arkov **R**asgele **A**lan
Bata	**B**ha**tt**acharyy**a**
FCN	**F**ully-**C**onnected **N**etwork
BoW	**B**ag **O**f **W**ords

Chapter 1

Introduction

Object and scene semantic segmentation (i.e. image parsing) are basic problems of computer vision. The goal of image parsing is segmentation of object and labeling of each object. In the last few years, complex and difficult classification methods became usable and accuracies of classification are really improved by helping the developments in the deep learning and more capacity in computation. Recent trends in image parsing are using superpixel based segmentation or training deep CNN architectures. We will explain these developments in the next sections.

1.1 Superpixels

Superpixel-based methods recently have the best performance in image labeling/parsing problems. Superpixels refer a limited form of region segmentation and combination of pixels. Their aim is reducing the complexity of image by using group of pixel for avoiding undersegmentation. [1] The image is segmented to atomic regions that are visually meaningful in superpixel-based segmentation method. These regions are coherent with object boundaries. Then the parsing/labeling algorithm applies to all the superpixel's pixels same semantic label. So, the process achieves a labeling of the entire image that is spatially consistent. However, superpixels provide attractive representation for computer vision problem by

Figure 1.1: Superpixel Segmentation [3]

representing redundancy in the image and describing original segment-based descriptors. Superpixels use morphological and geometric features of each segment beside of standard descriptors such as SIFT. [2]

In superpixel level, features are determined by Bag of Words (BoW) approach. First of all, feature vector is calculated for each pixel that belongs to superpixel. Then these features quantized as a vector and quantization indices histograms are extracted. The computed histogram is used as a feature descriptor for superpixel. [3]

1.2 Neural Network

Deep learning is one of the machine learning class. It uses many layers consecutively for extraction of feature. The algorithms are based on the supervised or unsupervised learning of features [4].

As shown in **Figure (1.2)** each layer is processed one by one and trained in a greedy manner consecutively. After the training of previous layers, a new layer is trained by using the previous layers that are input data of encoding. Then a supervised fine-tuning level of the entire network can be fulfilled.

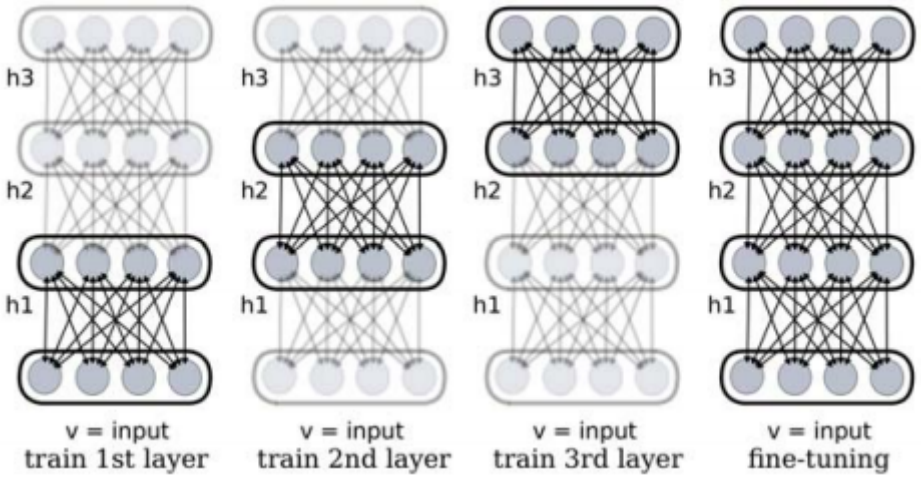


Figure 1.2: **Deep learning scheme:** the supervised fine-tuning stage that is affecting all layers are following a greedy unsupervised layer-wise pretraining stage [5]

Recently, in the image classification problems, deep learning methods such as convolutional neural network have obtained important popularity [6]. Actually, artificial neural networks are computational models that they are formed by considering the neuronal structure of the human brain [6]. There are lots of convolution and pooling layers in the CNN where their parameters are learned during the process of training [6]. Due to CNN usage, performance of computer vision tasks are improved such as image recognition [7], scene recognition [8], and video classification [9].

CNN consist of many layers that they are convolution layer, pooling layer, and fully-connected layer. In the convolution layer, it extracts features by using convolution operation. Pooling layer reduces feature dimension. In the fully-connected layer, input pattern is classified with high-level feature extracted from previous layers.

Generally deep networks compute a general nonlinear function of the whole image whereas Fully Convolutional Network (FCN) performs a nonlinear filtering of the image. [10]. The initial works on CNN use sliding window approach for semantic segmentation, but FCN uses multiple convolution layers to obtain dense pixel level labeling.

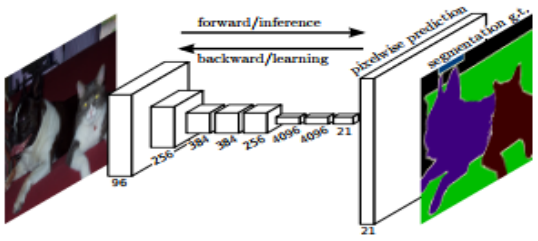


Figure 1.3: System of FCN [10]

1.3 Feature Encoding

Superpixel features are typically computed by Bag of Words (BoW) method. The basic approach in BoW is to perform quantization by assigning the feature vectors to the nearest codewords in the codebook and to use histogram of the quantized features as a descriptor [3]. Recently, feature encoding is being used instead of BoW.

Three basic steps can be mentioned in classification problems [11]:

- Extraction of local (pixel-level) features
- Obtaining feature descriptor for image/object/superpixel by using pooling and encoding of feature vectors
- Classification of image/object/superpixel descriptor

The basic method is to quantized local features for computing a spatial histogram of visual words. Alternative encoding methods have begun to replace the hard mapping method. This method keep more information about the original image features. This work can be done in two ways:

1. by representing features as combinations of visual words that is using by soft mapping [12] and local linear encoding [13], and

2. by saving the difference between the visual words and the features that is using by Fisher encoding [14] and super-vector encoding [15].

Recently, in the area of computer vision works are used soft mapping instead of hard mapping. Some of these works: Locality-Constrained Linear Coding [13], Fisher-vector Coding [14], Super-Vector Coding [15], Kernel Codebook Encoding [12]. Instead of hard mapping methods, using these improved coding methods reduce the loss of information caused by quantization to obtain more diverse descriptor. As a result, it provides improved classification results.

In this thesis, we focused on two of those encoding methods. One is KCB which uses soft mapping method instead of hard mapping for vector quantization. This mapping method assigns more than one codewords instead of assigning only one codeword [3]. The other one is LLC, which also uses soft mapping but with a different weighting mechanism.

1.4 Related Work

SuperParsing [16] represents an effective and simple nonparametric approach to the image segmentation problem. The approach needs no training, and it can easily set to different datasets that they have tens of thousands of images and hundreds of labels. The approach works as a scene-level matching by using global image descriptors, superpixel-level by using local features and MRF optimization for combining neighborhood context.

David & Fergus [17] proposed a non-parametric solution to scene parsing by considering the work of [16]. They added two different operation:

- A principled and efficient method to learn per-descriptor weights which is minimizing classification error,
- Training set's context driven adaptation that is used for each query to improve performance on rare classes.

We applied some weights to the different descriptors, but we found these descriptors by testing different alternatives.

In work of George [18], per-pixel accuracy was improved with a nonparametric scene parsing approach. Firstly, it improves estimation of label likelihood at superpixels by combining scores of likelihood from different probabilistic classifiers. Secondly, it was combining semantic context in the process of parsing through global label costs. The approach uses global likelihood cost estimate for each label, instead of likelihoods estimated from a retrieval set.

In Nguyen's adaptive [19] work, they present an adaptive nonparametric approach to the image parsing. For given test images, retrieval set is found from training images set. Then, each superpixel's category is assigned by the majority vote of the k-nearest neighbor superpixels. The method proposes a different adaptive nonparametric approach that it determines the sample specific k for each test image instead of fixing the k as same with traditional approaches of nonparametric methods.

In Nguyen's exploiting [6] work, they added the application of generic multi-level CNN approach into the image parsing. They were created the retrieval set by using global-level CNN feature matching similarities. Then, the similar images and the input test image are oversegmented into superpixels. Each superpixel's category is assigned by the majority vote of the k-nearest neighbor superpixels based on hand-crafted features and regional-level CNN features matching.

In work of Myeong & Lee [20] presents an original nonparametric approach for semantic segmentation by using high-order semantic relations. They propose the semantic relation transfer that is a method to move objects' high-order semantic relations from annotated images to unlabeled images.

1.5 Organization of Thesis

This thesis is organized as follows. In Chapter 2 we will explain superpixels and detailed of superparsing algorithm. Then feature encoding methods will be in Chapter 3. In Chapter 4 we will see more information about convolutional neural network. Our experiment details and results will be in Chapter 5 and we conclude our work and discuss future work in Chapter 6.

Chapter 2

SuperParsing Algorithm

One of the simple and efficient method for problem of image parsing is SuperParsing algorithm. The algorithm uses superpixels for image parsing. SuperParsing algorithm is based on nonparametric, data-driven approach by using superpixels for image parsing. It can easily be adapted to larger datasets and sets of label. The method does not require a classifier training instead it uses a retrieval set of scenes. The retrieval set is created from similar training image for each new test image. Retrieval set gives a knowledge for classification. It is found with matching at the level of scene by using global image features and then image is labeled with matching at the level of superpixel by using local features. If the retrieval set images and test images are very similar to each other, the labeling is transferred to the level of superpixels or coherent image regions generated by a bottom-up segmentation method. The label transfer works with an algorithm to find nearest-neighbors.

SuperParsing algorithm uses class conditional likelihood ratios that are computed by using local probability densities of features that are estimated from the training set. Generally, this approach gives successful results for datasets with medium to large sizes. However, estimation accuracy is not successful in the tails of the feature distribution because there are fewer samples of training for the rare, under-represented classes of the dataset. This problem can be solved by adding new training samples for the under-represented classes [17]. However, when setting the

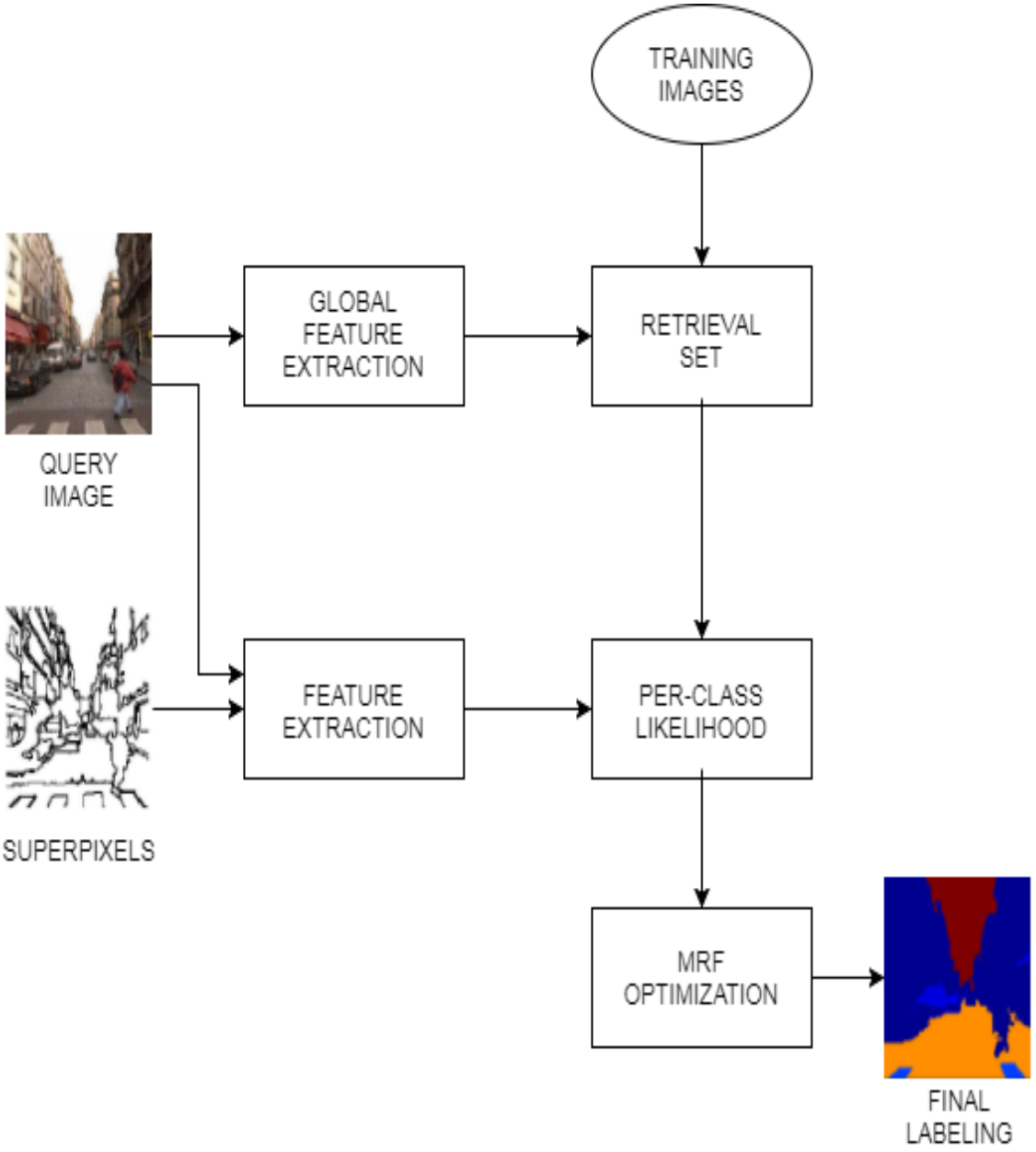


Figure 2.1: System Overview of SuperParsing Algorithm

dataset for improving mean class accuracy in labeling, generally reduces accuracy of pixel-level.

The system of this method works as shown in **Figure (2.1)**. Firstly, superpixels are generated from test image. At the same time, retrieval set is created as a

subset of similar images in the training set of test images. Feature extraction is performed for all the superpixels. Per-class likelihood is calculated by using feature distributions estimated from the retrieval set. Finally, Markov Random Field (MRF) optimization is performed on the image to get the final labeling.

2.1 Superpixels

Recently, scene and object classification are basic problems at the working on computer vision. Superpixels are successfully used for image labeling. Superpixels' form is cluster of multiple pixels, so they are more informative than pixels for scene analysis. Superpixels are obtained by using the fast graph-based segmentation algorithm of [21] and superpixels' appearance is described by using 20 different superpixel features. There are five features:

- **Shape:** It extracts the area of superpixel associated with image's area, the mask of superpixel through the bounding box of superpixel, and the bounding box width/height associated with image width/height to represent the shape [6].
- **Location:** It extracts the mask of superpixel shape through the image, and the bounding box's top height associated with image height [6].
- **Texture:** One of the important component in scene understanding is texture. Texton histogram [22], and the quantized SIFT histogram [16] are used as texture features.
- **Color:** The other important component is color in the human visual system for specifying properties of objects, scene understandings, etc. Superpixels help to extract color information such as RGB color information (mean,standard deviation), the color histogram (RGB, 11 bins per channel) [6].

- **Appearance:** Overall appearance of the superpixel can be represented by gist descriptor [23]. The grayscale gist is extracted from the bounding box of the superpixel [6].

Specially, texton histograms and dense SIFT descriptors are computed both for the superpixel region and for 10 pixels dilated region. Actually, the SIFT features are more powerful than texton features. Also, computing the left/right/top/bottom boundary histograms is useful for SIFT features. For each superpixel, all these features are computed and kept together with their class labels. A training superpixels is assigned to a class label if more than 50% of the superpixel region overlaps overlaps with the segment mask for that label.

2.1.1 Retrieval Set

The SuperParsing method is data-driven and nonparametric because of this retrieval set is used in the process. The step is important for the parsing method to find a good and comparatively small retrieval set of training images. Superpixels of a test image are matched with the training superpixels in this retrieval set only. This improves computational efficiency and provides scene level context for the subsequent superpixel matching step. A good retrieval set should iclude images of a similar scene type of training image together with similar objects and spatial layouts. To get this type of similarity between test image and training image, spatial pyramid, gist and color histogram can be used.

- **Spatial Pyramid:** This technique divides the image into gradually fine sub-regions and computing local features' histograms through the resulting sub-regions [24]. Also, the technique shows that the best results can be obtained by using combining of multiple resolutions.

- **Gist:** Observer can describe a meaningful information from a instantaneous scene that is a gist. The description of gist contains the scene semantic label

that are related to the function of scenes, some objects, and their surface characteristics, together with spatial layout [25].

- **Color Histogram:** It has a 3-channel RGB and there are 8 bins per each channel.

Euclidean distance of each feature type is calculated from the query images and all training images. Then all training images are ranked according to this calculation in increasing order. After that minimum of the each feature ranks is taken to obtain a single ranking for each training image. Finally, the top X that depends on dataset size, images are chosen as a retrieval set. To take the minimum of each feature ranks amounts gives better results than average of the ranks. Finding the best scene matches with good global descriptors is important for improving the success of subsequent superpixel matching. After the global matching, the retrieval set is ready to be an input for calculation of per-class likelihood for superpixel features encoded features which are coming from superpixels.

2.1.2 Local Superpixel Labeling

After segmenting the test image and extracting the superpixel features, a likelihood ratio score is computed for all possible class labels and for each feature of a test superpixel. For the given class, features are independent of each other by making the Naive Bayes assumption. For superpixel s_i and class c, the likelihood ratio is

$$L(s_i, c) = \log \frac{P(s_i|c)}{P(s_i|\bar{c})} = \sum_k \log \frac{P(f_i^k|c)}{P(f_i^k|\bar{c})} \tag{2.1}$$

where $\bar{c}$ is the set of all classes excluding c, and f_i^k is the feature vector of the k^{th} type for s_i. The conditional densities $P(f_i^k|c)/P(f_i^k|\bar{c})$ are locally estimated in the neighborhood of f_i^k using labeled feature vectors from the retrieval set [2]. Specifically, if D is the set of all superpixels in the training set and N_i^k is the set

of superpixels from the retrieval set within a local neighborhood of f_i^k, then

$$\frac{P(f_i^k|c)}{P(f_i^k|\bar{c})} = \frac{(n(c, N_i^k) + \epsilon)/n(c, D)}{(n(\bar{c}, N_i^k) + \epsilon)/n(\bar{c}, D)} = \frac{n(c, N_i^k) + \epsilon}{n(\bar{c}, N_i^k) + \epsilon} \times \frac{n(\bar{c}, D)}{n(c, D)} \tag{2.2}$$

where $n(c, S)$ is the number of superpixels in set S with class label c, and ϵ is a small constant used to smooth likelihood counts [2]. The set N_i^k contains retrieval set feature vectors whose L_2 distance from f_i^k is below a fixed threshold [2]. Note that the whole training set D is used to estimate $P(c)$ and $P(\bar{c})$, instead of just the retrieval set [2]. At that time, labeling of the image is obtained by simply assigning to each superpixel the class that maximizes **eq. 2.2**. After this initial labeling, contextual inference is performed in superpixel neighborhood using MRF optimization to smooth out and improve superpixel labels.

Chapter 3

Feature Encoding Methods

Superpixel features are typically computed by Bag of Words (BoW) method. The basic approach in BoW is to perform quantizition by assigning the feature vectors to the nearest codewords in the codebook and to use histogram of the quantized features as a descriptor [3]. Recently, feature encoding is being used instead of BoW. The basic method is to quantize local features for computing a spatial histogram of visual words. In the literature, there are two main types of mapping methods:

- **Hard Mapping:** The method matches nearest visual word in the given dictionary with each descriptor vector. However, two different feature vectors can be assigned same visual word without any distinction. This condition leads significant loss of information because of the codeword uncertainty that is not interested with the meaning ambiguity. Codeword uncertainty scales the contributions to each bin, so that each feature contributes a constant amount across all of the bins. In this method descriptor belongs to only one nearest word. It occurs lots of quantization fault because of these two reasons.

  As we mentioned on introduction *kmeans* is a clustering algorithm that is used to obtain the codewords in the codebook.

– ***Kmeans* Clustering:** Separates the local descriptor space into informative regions [11]. These regions are named as visual words and gathering of visual words is named as a visual vocabulary.

- **Soft Mapping:** The method assigns each descriptor vector to more than one nearest visual word. It sets weighted assignment according to word centers in local descriptor space. It keeps more information about original image features than hard mapping. Assigning weighted assignment to all visual words reduce problems of hard mapping .

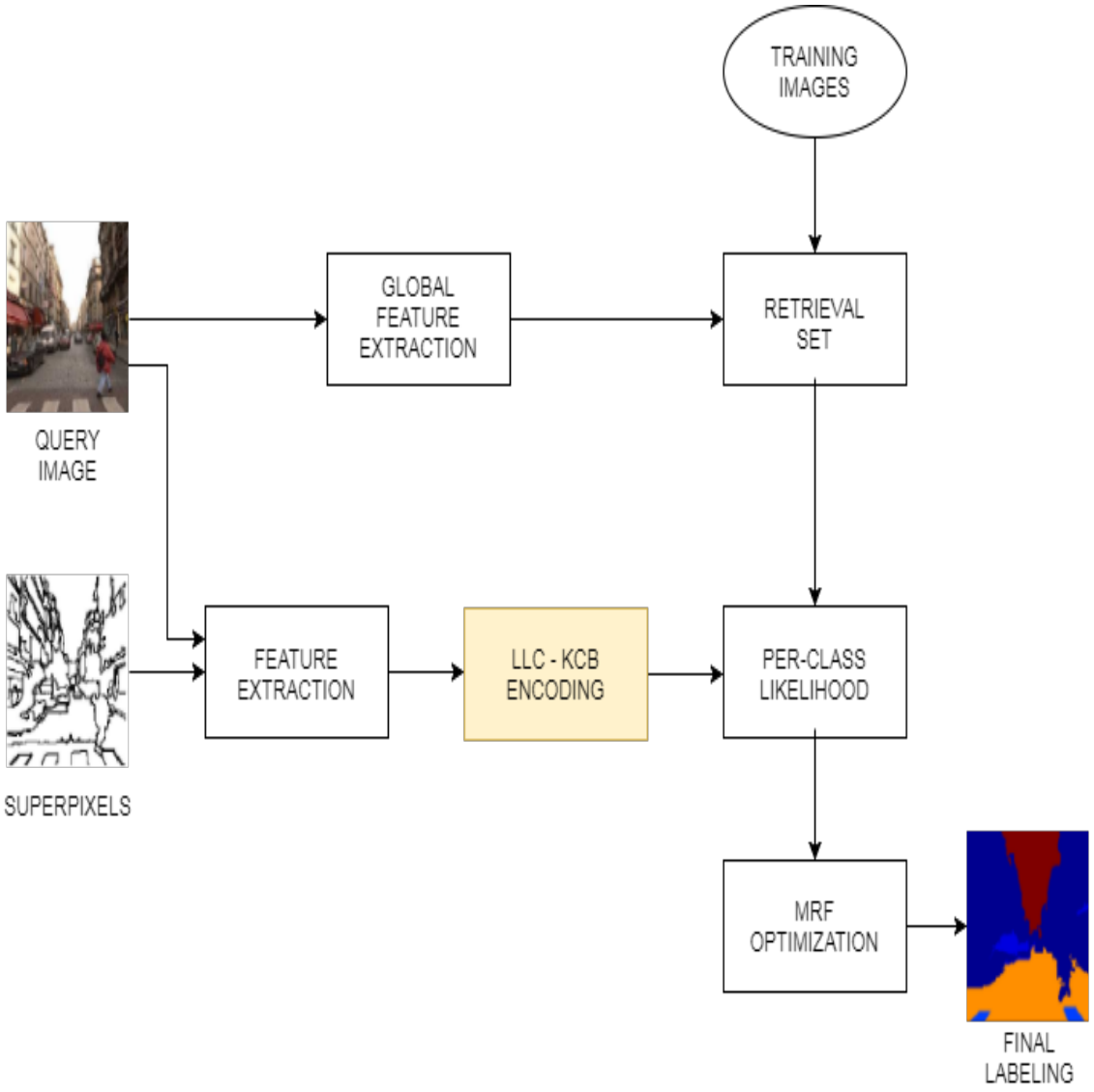


Figure 3.1: System Overview of Feature Encoding Methods

3.1 Kernel Codebook Encoding

KCB is the method that is applied soft mapping. Weights which are assigning to codewords, calculated by using kernel distance function. In [12], there are more information about coding of descriptors. The detailed information is like that ($\mathbf{x}$ is feature vector and let $\mathbf{a}_t$ be a codeword, $1 \leq t \leq A$) [3]:

$$[f_{kcb}(\mathbf{X})]_t = \frac{K(\mathbf{x}, \mathbf{a}_t)}{\sum_{j=1} K(\mathbf{x}, \mathbf{a}_j)} \tag{3.1}$$

when

$$K(\mathbf{x}, \mathbf{a}) = \exp(-\frac{\|\mathbf{x} - \mathbf{a}\|^2}{2\sigma^2}) \tag{3.2}$$

To get more efficient results on KCB encoding, the feature vectors should be recommended be encrypted with nearest 5 visual words. σ parameter determines how homogenous the weights are distributed in KCB. The weights get close each other when σ increases. In summary, shape of kernel directly comes from distance function, size of kernel depends on data and image descriptor.

Let T be a number of total visual word in codebook. Coefficient of features are zero except result of KCB encoding $\mathbf{f}_{kcb}(\mathbf{x}) = [[f_{kcb}(\mathbf{x})]_1 ... [f_{kcb}(\mathbf{x})]_T]$ length of the code vector is T and $A = 5$ nearest visual word.

The last step for KCB descriptor is merging of code vectors that they belong to superpixel. The most preffered methods are average pooling and maximum pooling.

- **Average Pooling:** The method decreases the dimension of data. Pooling is necessary to obtain single descriptor for the whole region. This operator sums and normalizes coefficients of all local descriptors that belong to the superpixel.

$$\mathbf{f}_{kcb} = \frac{1}{|SP|} \sum_{i \epsilon SP} \mathbf{f}_{kcb}(\mathbf{x}_i) \tag{3.3}$$

where $|SP|$ represents the size of superpixel that is the number of pixel. Hence, feature histogram is obtained by using average pooling method.

- **Maximum Pooling:** Also, this method works in the same way as an average pooling to decrease the dimension. The method choses the biggest coefficients to get the best classification performance. It represents whether each visual word exists or not in the image. In the encoding each bin is assigned to the maximum of SIFT feature encodings in that region.

3.2 Locality-Constrained Linear Encoding

LLC is the simple and effective coding method which is using soft quantization operation. It uses the locality constraints for projecting each descriptor into its local-coordinate system [13]. Then the projected coordinates are combined by using max pooling operation for creating the final representation.

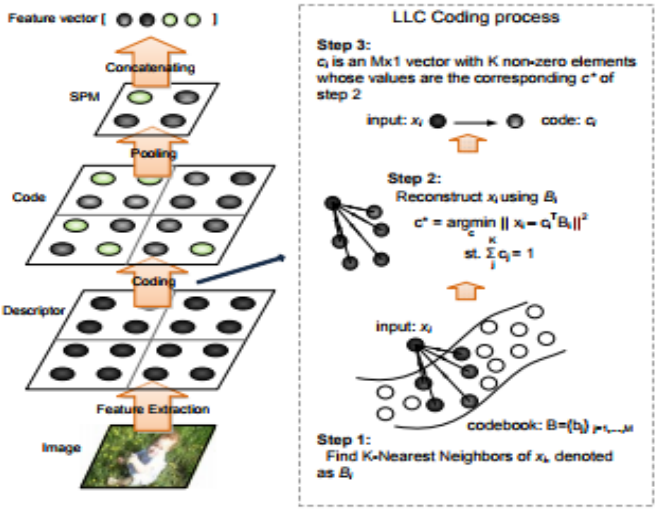


Figure 3.2: In image classification, spatial pyramid structure's flowchart for pooling features (left)
Locality-constrained linear coding process (right) [13]

In LLC, feature vector is expressed as a linear combination of codewords in the local neighborhood. A different approach is used to determine weights which are assigned to codewords. Assigned weights to the codewords $\alpha = [\alpha_1, \alpha_2, ..., \alpha_A]^T$ determines using the solution of following optimization problem:

$$\alpha^* = argmin_{1^T\alpha=1}\|\mathbf{x} - \mathbf{B}\alpha\|^2 + \beta\|\alpha\|^2$$

when x is a set of visual words closest to the feature vector, $B = [a_1, a_2, ..., a_A]$ column matrix occurs from the closest codewords, β is a small smoothing constant and sum of the weights is set equal to 1,that is $1^T\alpha = 1$.

If the number of visual word in the codebooks is T obtained from LLC and only the coefficients that correspond to $A = 5$ closest visual word, different from zero:

$$[f_{llc}(\mathbf{x})]_{I\mathbf{a}_t} = \alpha_t, \qquad 1 \leq t \leq A$$

($I\mathbf{a}_t$ is row number in the vocabulary of a visual word $\mathbf{a}_t$)

The last step of LLC is combining of superpixels codevectors. In LLC, it was observed that the maximum pooling was more successful than the average pooling. Because of that,

$$\mathbf{f}_{llc} = \max_{i\epsilon SP}[\mathbf{f}_{llc}(\mathbf{x}_i)]_t, \qquad 1 \leq t \leq T$$

Finally, superpixel codevector is normalized to be norm 1 as following:

$$\mathbf{f}_{llc}^{\mathbf{N}} = \frac{\mathbf{f}_{llc}}{\|\mathbf{f}_{llc}\|}$$

3.3 Modifications of Feature Encoding

We use same global features as SuperParsing, which are SpatialPyramid, Gist of Color, Color Histograms. We also use the same local descriptors as SuperParsing except for six SIFT descriptor. These SIFT descriptors are either KCB or LLC encoded and used in superpixel matching. Therefore the modifications are as follows:

- We applied the encoding methods (LLC - KCB) instead of Bag of Words method for SIFT features.
- Nearest R number superpixels from retrieval set are included to N_i^k neighborhood.
- L_1 and *Bhattacharyya* (d_B) metrics are tested for distance computation between KCB encoded SIFT descriptors

$$d_B(\mathbf{f}_i, \mathbf{f}_j) = -ln\left(\sum_{t=1}^{T}\sqrt{[f_i]_t.[f_j]_t}\right)$$

- L_2 distance is used for LLC encoded SIFT descriptors.

Chapter 4

Convolutional Neural Networks

Convolutional Neural Networks are known as CNNs or ConvNets. Convolutional Neural Networks and Neural Networks are similar to each other. Neural Networks consist of neurons which are able to learn weights and biases. Some inputs are received from each neuron, these neurons perform a dot product on received inputs and follows it with a non-linearity. The all network refer a single differentiable score function that is from the pixels of raw image on one end to class scores at the other. Also, the neural networks have a loss function on the last layer that is fully-connected. [26]

Convolutional Neural Network architectures make the clear assumption that the inputs are images, segment descriptors and global descriptors. The ConvNet architectures let us encode exact properties into the architecture. After that, these make the forward function more efficient for implementing and reducing the amount of parameters in the network.

4.1 Overview

Neural Networks receive a single vector as an input, and convert it by using a series of hidden layers. Each hidden layer consists of a series of neurons. These neurons are fully-connected to all neurons in the previous layer, they work completely independently in a single layer function. Also, they do not share any connections.

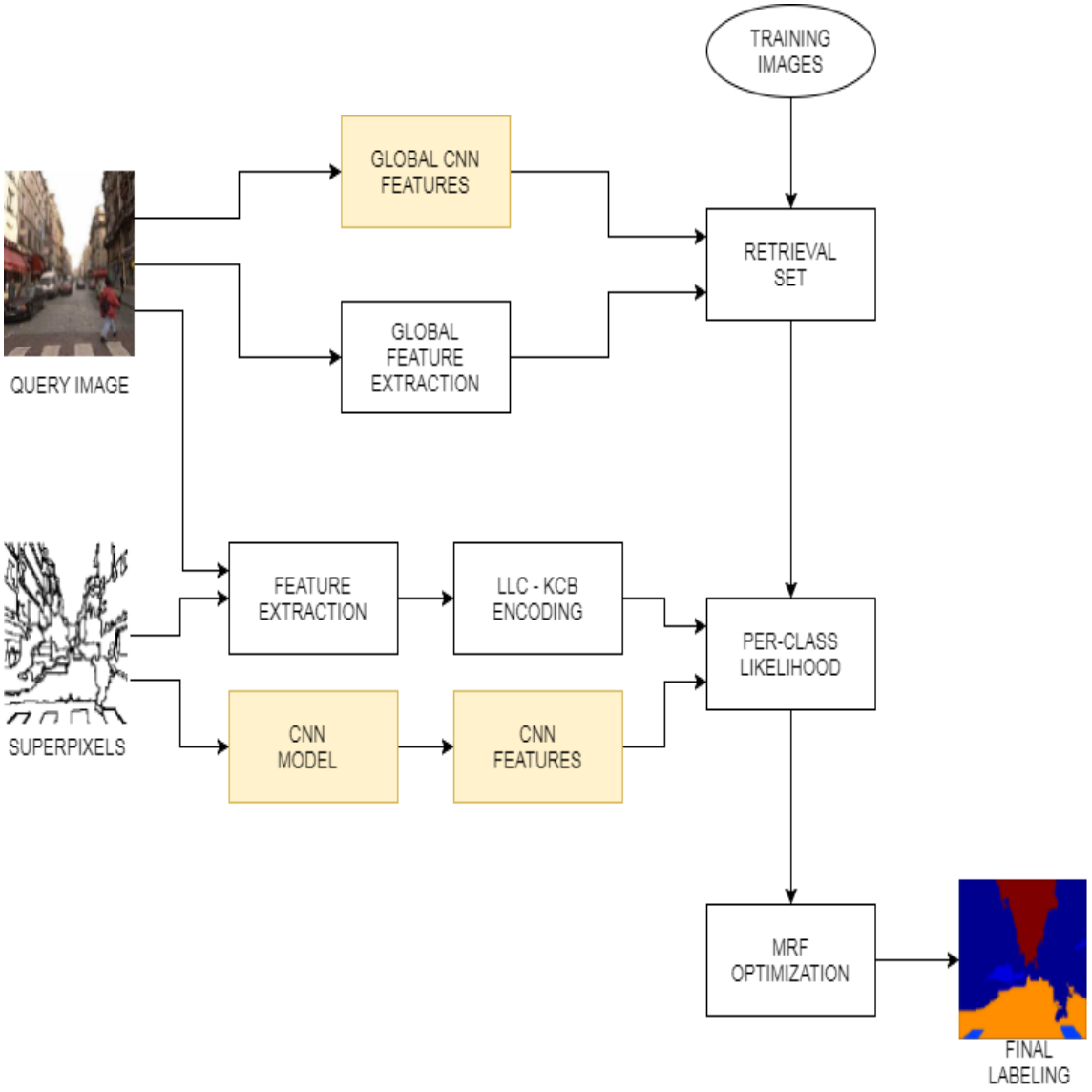


Figure 4.1: System Overview of CNN Models

The last layer that is fully-connected layer, is named output layer and it shows the class scores in classification settings. [26]

Convolutional Neural Networks benefit from the fact that the input is an image. Especially, the layers of a CNNs have arrangement of neurons in 3 dimensions: width, height, and depth differently according to Neural Network.

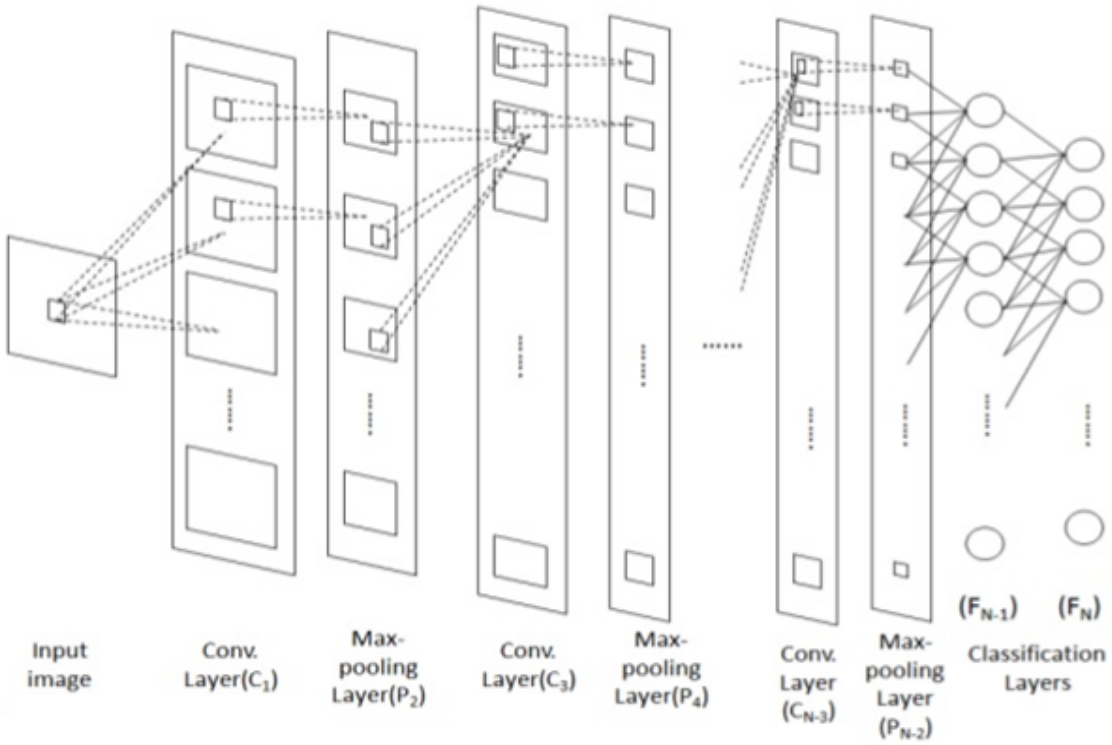


Figure 4.2: System Overview of Convolutional Neural Network Layers [27]

4.2 ConvNet Layers

A Convolutional Neural Networks consist of layers. Each layer converts an input volume to an output volume by using a differentiable function. There are three main types of layers for building ConvNet architectures:

- Convolutional Layer
- Pooling Layer
- Fully-Connected Layer

All these layers can be seen in **Figure (4.2)** as an example. First of all, we extract the feature in the convolution layers and then reduce the feature dimension by using maximum pooling. Finally, we use the last layer for classifying.

4.2.1 Convolutional Layer

The convolutional layer is the main part of a ConvNet which is lifting the heavy computation. In this layer, nodes are grouped on each layer into 2-D planes. Each plane is connected with more than one input planes. In a small region, weighted sum of input nodes is computed by each node.

4.2.2 Pooling Layer

Generally, inserting a pooling layer to a successive convolutional layers in a ConvNet architecture is used. It is used to reduce the quantity of parameters and computation in the network by reducing the spatial size of the representation as shown in **Figure (4.3)**. Because of these, it also controls overfitting.

In the pooling layers that is generally called max-pooling layers, nodes are grouped on each layer into planes. Differently from convolution layer, each plane is connected with just one input plane. In a small region, each node chooses maximum from the input nodes. [27]

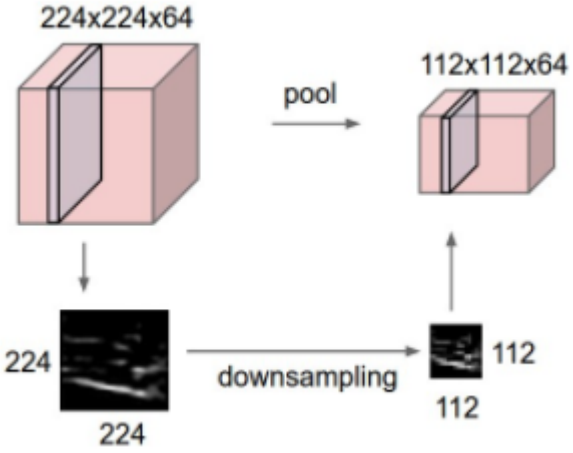


Figure 4.3: Pooling layer downsamples the volume spatially, the volume depth is preserved [26]

The max operation at the pooling layer operates independently on the input's each depth slice and changes the slice size spatially, see **Figure (4.4)**. Zero-padding usage is not common for pooling layers.

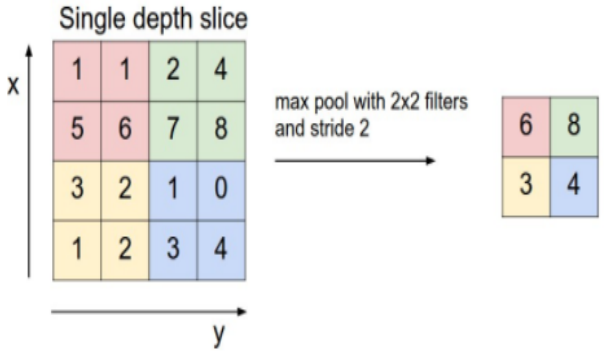


Figure 4.4: Example of max operation [26]

4.2.3 Fully-Connected Layer

The other most common layer in ConvNet is fully-connected layers is that these layers are identified as a family of functions. The weights of the network parameterize these layers. In this layer, single layer's neurons don't share connections, but between two adjacent layers' neurons are fully pairwise connected [28]. As shown in **Figure (4.5)**, all the input nodes are fully connected with each node. Every nodes compute sum of weighted of all the input nodes.

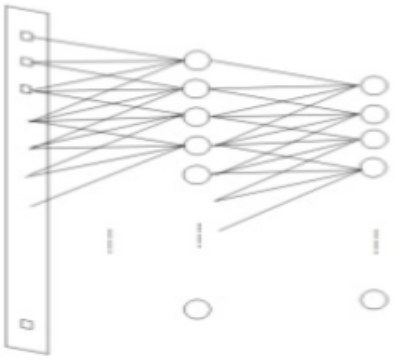


Figure 4.5: Fully-Connected Layer [27]

Fully-connected layer's neurons have connections with all activations in the previous layer, exactly like in regular Neural Networks. Hence, neurons' activations can be calculated by using a matrix multiplication. There are two example of fully-connected layers for Neural Network:

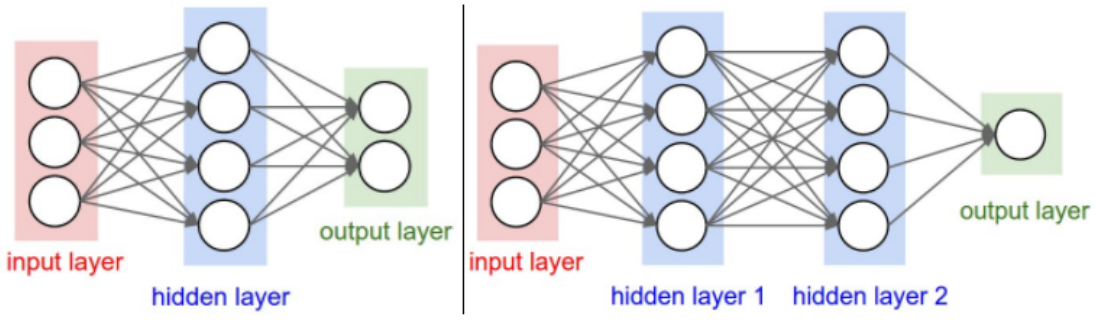


Figure 4.6: A 2-Layer Neural Network that consists 3 inputs, 1 hidden layer with 4 neurons, and 1 output layer with 2 neurons (left)
A 3-Layer Neural Network that consists 3 inputs, 2 hidden layers each with 4 neurons, and 1 output layer with 1 neuron (right) [26]

4.3 Feature Extraction using ConvNets

Deep neural networks learn image features in a hierarchy. Several works have shown before that these learned features can be used in image classification tasks instead of hand-crafted features. In deep neural networks, low level features, such as corners, edges, are learned from lower layers whereas color, shape etc. are learned by middle layers. Also, higher layers learn high level features which represent the objects in the image. [29]

Furthermore, CNN can be used as a feature extractor by using the activations available before network's last fully connected layer. The activations will be acting as a feature vector for a general classifier during learning and classification stages. The pre-trained CNN models such as imagenet-vgg and alexnet could be used as a feature extractor. [29]

4.4 Modifications of CNN

SuperParsing uses global descriptors such as Spatial Pyramid, Gist of Color, Color Histograms for global matching to find the retrieval set, and local descriptors such as SIFT, texton etc. Additionally to these descriptors CNN models such as imagenet-vgg [30] and alexnet [7], which are trained on ILSVRC ImageNet dataset [31], use both for global and local descriptors. Therefore the modifications are as follows:

- As an alternative to global descriptors of SuperParsing, we use learned features from alexnet and vgg for the global matching of the whole image.
- We also use alex and vgg features as local descriptors for superpixels for this purpose we set the whole image to zero except for the superpixel region and compute the output of the cnn model for this superpixel only image
- We added global and segment descriptors of CNN models in addition to LLC or KCB encoded features. Other part of the system remains the same.

Chapter 5

Experimental Work

In our work, we applied LLC, and KCB encoding methods, and CNN feature learned from pre-trained models on SIFT Flow and 19-class LabelMe datasets. Also, we used MRF optimization to improve resulting accuracy. MRF is a joint probability distribution's graphical model. It is used to make contextual inference by using superpixel neighborhoods.

We used the original SuperParsing segment descriptors which represent shape, location, texture, color, and appearance. In our work, we replaced the six original SIFT descriptors [16] with their KCB and LLC encoded versions. Also, we used learned CNN features from pre-trained CNN architectures such as alexnet and imagenet-vgg to improve pixel accuracy. Descriptors used for:

- Shape
  - 'centered_mask_sp'
  - 'bb_extent'
  - 'pixel_area'
- Location
  - 'absolute_mask'
  - 'top_height'

- Texture
  - 'int_text_hist_mr'
  - 'dial_text_hist_mr'
- Color
  - 'mean_color'
  - 'color_std'
  - 'color_hist'
  - 'dial_color_hist'
- Appearance
  - 'color_thumb'
  - 'color_thumb_mask'
  - 'gist_int'
- LLC
  - 'LLC_Sift'
  - 'LLC_Sift_dial'
  - 'LLC_Sift_bottom'
  - 'LLC_Sift_top'
  - 'LLC_Sift_right'
  - 'LLC_Sift_left'
- KCB
  - 'KCB_Sift'
  - 'KCB_Sift_dial'
  - 'KCB_Sift_bottom'
  - 'KCB_Sift_top'

  - 'KCB_Sift_right'
  - 'KCB_Sift_left'

- CNN
  - 'cnn_feat'
  - 'cnn_feat_alex'

- Global Descriptors
  - 'spatialPryScale'
  - 'colorGist'
  - 'coHist'
  - 'cnn_feat'
  - 'cnn_feat_alex'

We made our tests with different parameter set as follows. We used two different retrieval set size as 100 and 200 images. In original SuperParsing [16], they used 200 images for retrieval set. We used 100 images retrieval set for some 19-class LabelMe dataset experiments. In graph-based segmentation method (GBS) [21], superpixel color consistency is controlled by K and the minimum superpixel size is determined by S. The nearest neighbor set size is R. The average number of samples in the retrieval set and R is proportional to each other. Value of R is 15 and 30 when $(K = 400, S = 200)$ and $(K = 200, S = 100)$, respectively.

In addition to all, we tested two different alternatives of vocabulary size and β parameter for LLC, three different alternatives of σ parameter for KCB. Value of vocabulary sizes are 256, and 512. β parameters' values are 0.012, and 0.015. The σ parameters are 75, 100, and 125.

We used L_2-normalization for LLC feature vectors and L_1-normalization for KCB feature vectors [16]. We tested L_1 and *Bhattacharyya* metrics d_B to compute distance between KCB encoded SIFT descriptors. L_2 metric is used for LLC, original

descriptors, and CNN features. In simulations, pixel-level classification accuracy (i.e. correctly classified pixel percentage) and average of per-class accuracies are compared.

5.1 SIFT Flow Dataset

The dataset consists of $2,688$ images with these 33 classes:

1. Sky
2. Building
3. Tree
4. Mountain
5. Road
6. Sea
7. Field
8. Car
9. Sand
10. River
11. Plant
12. Grass
13. Window
14. Sidewalk
15. Rock
16. Bridge
17. Door
18. Fence
19. Person
20. Staircase
21. Awning
22. Sign
23. Boat
24. Crosswalk
25. Pole
26. Bus
27. Balcony
28. Streetlight
29. Sun
30. Bird
31. Cow
32. Dessert
33. Moon

These images size are $256x256$ pixels. Generally, the dataset is divided into $2,488$ images for training and 200 images for test.

5.1.1 Experiments with LLC

Shape, location, texture, color, appearance, and LLC encoded descriptors are used for LLC experiments. Original SIFT descriptors are removed. We used retrieval set size as 200 in this experiment. LLC tables are organized as follows:

MX1: X2/X3/X4

X1, X2, X3, X4 refer to model number, vocabulary size, β parameter, distance metric, respectively.

- **M1:** 256/0.015/L2
- **M2:** 512/0.012/L2

	Percentage Accuracy (%)	
	Base	MRF
Method	Per-pixel/Per-class	Per-pixel/Per-class
M1	76.7/28.6	77.6/24.6
M2	75.8/29.1	76.8/26.7

Table 5.1: K400_S200 with baseline and best MRF on SIFT Flow dataset for LLC

We tested LLC method with two different parameter sets that they are $K = 400$, $S = 200$ and $K = 200$, $S = 100$.

For the first set that is $K = 400$, $S = 200$:

We can see at the **Table 5.1**, using small vocabulary size (256) and big β parameter (0.015) can improve the per-pixel accuracy approximately %1. However, average per-class accuracies are dropped down with these parameter settings.

- **M3:** 256/0.015/L2
- **M4:** 512/0.015/L2

	Percentage Accuracy (%)	
	Base	MRF
Method	Per-pixel/Per-class	Per-pixel/Per-class
M3	76.4/33.1	77.9/30.9
M4	76.0/32.7	77.7/31.1

Table 5.2: K200_S100 with baseline and best MRF on SIFT Flow dataset for LLC

For the second set that is $K = 200$, $S = 100$:

We did the experiment with same β parameter (0.015), distance metric (L_2), and using different vocabulary size at the **Table 5.2**. In these tests, we can see the smaller vocabulary size (256) results are better than bigger vocabulary size (512) results. In these tests, per-class accuracies are less affected.

In summary for LLC encoding method, the best result is obtained when $K200\_S100$, vocabulary size is 256 and β is 0.015.

5.1.2 Experiments with KCB

Shape, location, texture, color, appearance, and KCB encoded descriptors are used for KCB experiments. Original SIFT descriptors are removed. We used retrieval set size as 200 in this experiment. KCB tables are organized as follows:

MX1: X2/X3/X4

X1, X2, X3, X4 refer to model number, vocabulary size, σ parameter, distance metric, respectively.

- **M5:** 256/75/L1
- **M6:** 512/75/L1
- **M7:** 256/75/Bata
- **M8:** 512/75/Bata

	Percentage Accuracy (%)	
	Base	MRF
Method	Per-pixel/Per-class	Per-pixel/Per-class
M5	77.1/29.9	77.6/29.0
M6	77.7/30.5	78.2/26.4
M7	77.1/30.1	77.9/27.5
M8	77.1/30.3	77.9/25.5

Table 5.3: K400_S200 with baseline and best MRF on SIFT Flow dataset for KCB

We tested two different distance metrics that they are L_1 and Bhattacharyya and two different parameter sets that they are $K = 400$, $S = 200$ and $K = 200$, $S = 100$.

For the first set that is $K = 400$, $S = 200$:

We did four experiments at the **Table 5.3** with same σ parameter (75) and two different vocabulary sizes that are 256 and 512. If we compare L_1 results, we can see that better per-pixel results are obtained when vocabulary size increases and per-class accuracy is nearly the same for baseline method but lower in MRF method. If we compare *bhattacharyya* results, we can see that nearly same results are obtained for baseline method and for MRF optimization when vocabulary size increases and per-class accuracy is nearly the same for baseline method but less in MRF method.

- **M9:** 256/100/L1
- **M10:** 512/100/L1
- **M11:** 256/100/Bata
- **M12:** 512/100/Bata

	Percentage Accuracy (%)	
	Base	MRF
Method	Per-pixel/Per-class	Per-pixel/Per-class
M9	77.2/33.2	78.4/29.6
M10	77.3/33.3	78.4/30.2
M11	76.9/33.8	78.3/30.7
M12	76.8/33.1	78.1/30.4

Table 5.4: K200_S100 with baseline and best MRF on SIFT Flow dataset for KCB

For the second set that is $K = 200$, $S = 100$:

We did four experiments **Table 5.4** with same σ parameter (100) and two different vocabulary sizes that are 256 and 512. If we compare L_1 results, we can see that nearly same results are obtained for baseline method and for MRF optimization when vocabulary size increases and per-class accuracies are also nearly the same for both methods. If we compare *bhattacharyya* results, we can see that

nearly same per-pixel results are obtained for both baseline methods and MRF optimization when vocabulary size changes and per-class accuracies are nearly the same for both methods.

In summary for KCB encoding method, the best result is obtained when $K200\_S100$, vocabulary size is 512 and σ is 100 with L_1 distance metric.

5.1.3 Experiments with CNN

We observed that, when CNN features are used, some of the descriptors of shape, texture, and color actually decreased the accuracy percentage. These descriptors such as 'bb_extent', 'int_text_hist_mr', 'dial_text_hist_mr', 'color_std', are ignored to improve accuracy. However, we added 'cnn_feat' and 'cnn_feat_alex' descriptors. KCB or LLC descriptors are also used instead of original SIFT descriptors for CNN experiments. We named these reduced set of descriptors as "limited". Also, we tested "full" version that is using all descriptors including CNN features and KCB or LLC encoded versions of SIFT descriptors. In these alternatives, we gave different weights to descriptors. For instance; we assigned weight 3 for 'KCB_Sift_top', 'KCB_Sift_bottom', and 'mean_color', weight 2 for 'cnn_feat' and 'cnn_feat_alex', and weight 1 for others. We named this weight set as a "ch". When all segment descriptors are one, it is called "1".

In addition to all these descriptor sets, we also changed global descriptors. We tested nearly all combinations between 'spatialPryScaled' (sps), 'colorGist' (cG), 'coHist' (cH), 'cnn_feat' (vgg), and 'cnn_feat_alex' (alex). We used retrieval set size as 200 in this experiment. CNN tables are organized as follows:

MX1: X2/X3

X1, X2, X3 refer to model number, which global descriptors are used, which segment descriptors are used, respectively.

- **M13:** cH_alex/limited/1
- **M14:** sps_alex/limited/1
- **M15:** sps_vgg/limited/1
- **M16:** cG_cH_alex/full/1
- **M17:** sps_cG_alex/full/1

	Percentage Accuracy (%)	
	Base	MRF
Method	Per-pixel/Per-class	Per-pixel/Per-class
M13	81.2/34.8	81.1/34.4
M14	81.6/35.7	81.5/35.0
M15	81.4/37.6	81.6/35.8
M16	81.0/32.6	81.1/30.9
M17	81.2/32.9	81.0/30.0

Table 5.5: K400_S200 with baseline and best MRF on SIFT Flow dataset for CNN

We tested five different alternatives with $K = 400$, $S = 200$ for CNN method. At the **Table 5.5** , we can see that limited experiments are more succesful than full experiments. It seems that the best global descriptor partner is SpatialPryScaled and imagenet-vgg or alexnet. Other alternative partners dont give the good results on per-pixel accuracy but SpatialPryScaled and imagenet-vgg partner gives best per-class accuracy. In the CNN tests MRF optimization doesn't achieve almost any improvement on per-pixel accuracy.

- **M18:** sps_alex/full/ch
- **M19:** sps_vgg/full/ch
- **M20:** cH_alex/full/ch

	Percentage Accuracy (%)	
	Base	MRF
Method	Per-pixel/Per-class	Per-pixel/Per-class
M18	79.9/36.0	80.4/38.3
M19	80.4/38.9	80.8/33.0
M20	80.1/39.5	80.6/31.3

Table 5.6: K200_S100 with baseline and best MRF on SIFT Flow dataset with LLC encoded for CNN

We tested three different alternatives with $K = 200$, $S = 100$ for CNN method. At the **Table 5.6** , we can see that full type of descriptors are the best experiments. It seems that the best global descriptor partner is SpatialPryScaled and imagenet-vgg. In the CNN tests MRF optimization doesn't achieve almost any improvement on per-pixel accuracy.

In summary for CNN models, the best result is obtained when $K400\_S200$, descriptors are limited and global descriptors are SpatialPryScaled and imagenet-vgg.

	Percentage Accuracy (%)	
	Base	MRF
Method	Per-pixel/Per-class	Per-pixel/Per-class
M1	76.7/28.6	77.6/24.6
M6	77.7/30.5	78.2/26.4
M15	81.4/37.6	81.5/35.0
SuperParsing [16]	74.1/30.2	76.2/29.1
DavidFergus [17]	76.8/39.2	77.1/32.5
George [18]	78.3/33.2	81.7/50.1

Table 5.7: Best results on K400_S200 for SIFT Flow dataset

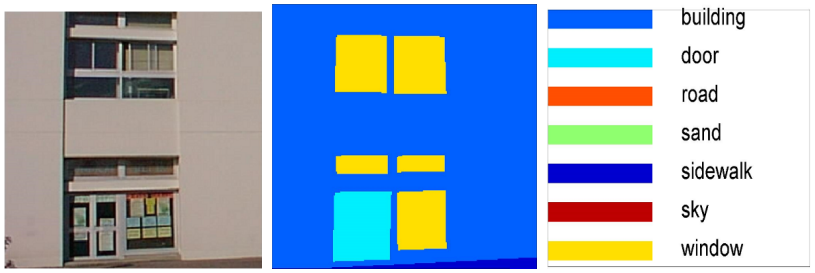
At the **Table 5.7**, we can see the best accuracy results on SIFT Flow dataset for $K400\_S200$. Pre-trained CNN features are more effective than feature encoding methods on parsing performance both for per-pixel and per-class accuracies. Our work is providing really good improvements over the original SuperParsing [16] and David & Fergus [17] who worked on learning per-descriptor weights and context driven adaptation. Also, our work gives nearly as good result as George [18] on per-pixel accuracy but the work of George, which uses classifiers trained in balanced datasets, is much better in terms of average per-class accuracy.

At the **Table 5.8**, we can see the best accuracy results on SIFT Flow dataset for $K200\_S100$. Again pre-trained CNN models are better than feature encoding methods on parsing performance both for per-pixel and per-class accuracies. $K200\_S100$ and $K400\_S200$ parameters' results are nearly same for feature encoding methods, but $K400\_S200$ results are better for CNN-based methods.

	Percentage Accuracy (%)	
	Base	MRF
Method	Per-pixel/Per-class	Per-pixel/Per-class
M3	76.5/33.1	77.9/30.9
M10	77.3/33.3	78.4/30.2
M20	80.1/39.5	80.6/31.3
SuperParsing [16]	74.1/30.2	76.2/29.1
DavidFergus [17]	76.8/39.2	77.1/32.5
George [18]	78.3/33.2	81.7/50.1

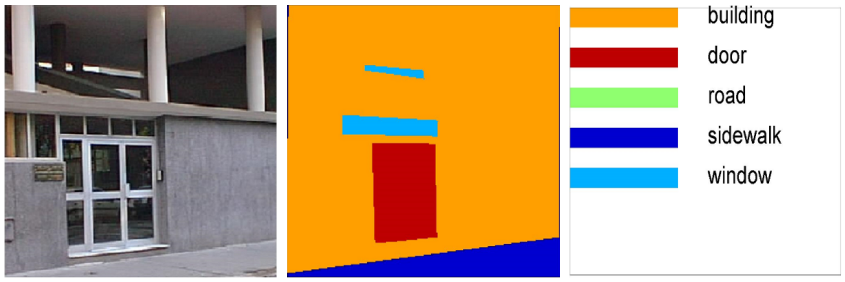
Table 5.8: Best results on K200_S100 for SIFT Flow dataset

In summary, when using LLC encoding method, we saw 2.6% difference with baseline method on per-pixel accuracy for $K400\_S200$. When we applied for $K200\_S100$, accuracy is increasing approximately 2.5% over original SuperParsing. When using KCB encoding method, we saw 3.6% difference with baseline method on per-pixel accuracy for $K400\_S200$. When we applied for $K200\_S100$, accuracy is increasing approximately 3.2%. When using CNN model, we saw 7.3% difference with baseline method on per-pixel accuracy for $K400\_S200$. When we applied for $K200\_S100$, accuracy is increasing approximately 6%.



Original Image Ground Truth Legend

Method M3 Method M9 Method M14



Original Image Ground Truth Legend

Method M3 Method M9 Method M14

Figure 5.1: Visual representation of best results of SIFT Flow dataset

We applied three best methods to the original images. Also, visual representations give ground truth and class legends. Information of these methods as follows:

- M3:
  - LLC encoding
  - Vocabulary size: 256
  - β: 0.015
  - Distance metric: L_2
  - Superpixel color consistency (K): 200
  - Minimum superpixel size (S): 100
- M9:
  - KCB encoding
  - Vocabulary size: 256
  - σ: 100
  - Distance metric: L_1
  - Superpixel color consistency (K): 200
  - Minimum superpixel size (S): 100
- M14:
  - CNN encoding
  - Used global descriptors: Spatial Pyramid & alexnet
  - Weights of descriptros: ch
  - Type of segment descriptors: limited
  - Superpixel color consistency (K): 400
  - Minimum superpixel size (S): 200

For the first original image, method 14 assigns road, building and window most correctly. For the second original image, method 14 assigns road, building, window, and door more detailed than others. In summary, generally $K = 200$,

$S = 100$ combination and smaller vocabulary size give better results but when CNN models are used for segment and global descriptors, they give the best results.

5.2 19-Class LabelMe Dataset

The dataset consists of 350 images from the LabelMe dataset. These images are chosen randomly. There are 19 classes:

1. Sky
2. Building
3. Tree
4. Mountain
5. Road
6. Field
7. Car
8. Sand
9. Grass
10. Sidewalk
11. Rock
12. Person
13. Sign
14. Boat
15. Water
16. Ground
17. Bison
18. Snow
19. Airplane

These images size are always $640x480$ pixels, some of the images are differently in pixel size. Generally, the dataset is divided 250 images for training and 100 images for test.

5.2.1 Experiments with LLC

Shape, location, texture, color, appearance, and LLC encoded descriptors are used for LLC experiments. Original SIFT descriptors are removed as in SIFT Flow dataset experiments. We used retrieval set size as 100 in this experiment. LLC tables are organized as follows:

MX1: X2/X3/X4

X1, X2, X3, X4 refer to model number, vocabulary size, β parameter, distance metric, respectively.

- **M21:** 256/0.015/L2
- **M22:** 512/0.015/L2

	Percentage Accuracy (%)	
	Base	MRF
Method	Per-pixel/Per-class	Per-pixel/Per-class
M21	78.9/49.7	79.5/49.9
M22	78.6/49.7	79.8/49.9

Table 5.9: K400_S200 with baseline and best MRF on 19-class LabelMe dataset for LLC

We tested LLC method with two different parameter sets that are $K = 400$, $S = 200$ and $K = 200$, $S = 100$.

For the first set that is $K = 400$, $S = 200$:

We did the experiment by using same β parameter (0.015) We can see at the **Table 5.9**, using both vocabulary size (256 and 512) have nearly same per-pixel and per-class accuracy results.

- **M23:** 256/0.015/L2
- **M24:** 512/0.015/L2

	Percentage Accuracy (%)	
	Base	MRF
Method	Per-pixel/Per-class	Per-pixel/Per-class
M23	80.7/55.0	82.7/55.0
M24	81.3/55.6	82.9/55.5

Table 5.10: K200_S100 with baseline and best MRF on 19-class LabelMe dataset for LLC

For the second set that is $K = 200$, $S = 100$:

We did the experiment with same β parameter (0.015), distance metric (L_2), and using different vocabulary size at the **Table 5.10**. In these tests, we can see the bigger vocabulary (512) size results are better than smaller vocabulary size (256) results. In these tests, per-class accuracy did not change too much.

In summary for LLC encoding method, the best result is obtained when $K200\_S100$, vocabulary size is 512 and β is 0.015.

5.2.2 Experiments with KCB

Shape, location, texture, color, appearance, and KCB encoded descriptors are used for KCB experiments. Original SIFT descriptors are removed as in SIFT Flow dataset experiments. We used retrieval set size as 100 in this experiment. KCB tables are organized as follows:

MX1: X2/X3/X4

X1, X2, X3, X4 refer to model number, vocabulary size, σ parameter, distance metric, respectively.

- **M25:** 256/100/L1
- **M26:** 512/100/L1

	Percentage Accuracy (%)	
	Base	MRF
Method	Per-pixel/Per-class	Per-pixel/Per-class
M27	78.4/48.7	79.0/48.2
M28	78.5/49.1	79.2/48.8

Table 5.11: K400_S200 with baseline and best MRF on 19-class LabelMe dataset for KCB

We tested KCB method with L_1 distance metric and two different parameter sets that they are $K = 400$, $S = 200$ and $K = 200$, $S = 100$.

For the first set that is $K = 400$, $S = 200$:

We did two experiments at the **Table 5.11** with same σ parameter (100) and two different vocabulary sizes that are 256 and 512. We can see that better per-pixel and per-class accuracy results are obtained when vocabulary size increases.

- **M29:** 256/100/L1
- **M30:** 512/75/L1

	Percentage Accuracy (%)	
	Base	MRF
Method	Per-pixel/Per-class	Per-pixel/Per-class
M29	80.7/55.7	82.5/55.0
M30	80.5/55.8	81.5/53.4

Table 5.12: K200_S100 with baseline and best MRF on 19-class LabelMe dataset for KCB

For the second set that is $K = 200$, $S = 100$:

We did two experiments **Table 5.12** with two different σ parameter (75 and 100) and two different vocabulary size that they are 256 and 512. We can see that better per-pixel results are obtained with smaller vocabulary size, bigger σ parameter value.

In summary for KCB encoding method, the best result is obtained when $K200\_S100$, vocabulary size is 256 and σ is 100.

5.2.3 Experiments with CNN

KCB descriptors are used instead of original SIFT descriptors for CNN experiments. We observed that some of the descriptors in shape, texture, and color decreased the accuracy percentage. These descriptors such as 'bb_extent', 'int_text_hist_mr', 'dial_text_hist_mr', 'color_std', are ignored to improve accuracy. However, we added 'cnn_feat' and 'cnn_feat_alex' descriptors. We named these type of descriptor set is "limited". Also, we tested "full" version that is used all descriptors without original SIFT and LLC descriptors. In these alternatives, we gave different weights to descriptors. For instance; we assigned weight 3 for 'KCB_Sift_top', 'KCB_Sift_bottom', and 'mean_color', weight 2 for 'cnn_feat' and 'cnn_feat_alex', and weight 1 for others. We named this weight set as a "ch". When all segment descriptors are one, it is called "1".

In addition to all these descriptor sets, we also changed global descriptors. We tested nearly all combinations between 'spatialPryScaled' (sps), 'colorGist' (cG),

'coHist' (cH), 'cnn_feat' (vgg), and 'cnn_feat_alex' (alex). We used retrieval set size as 200 in this experiment. CNN tables are organized as follows:

MX1: X2/X3/X4

X1, X2, X3, X4 refer to model number, which global descriptors are used, which weight is assigned, which segment descriptors are used, respecctively.

- **M31:** sps_vgg/ch/limited
- **M32:** sps_alex/ch/limited
- **M33:** cH_alex/ch/limited
- **M34:** cG_cH_alex/1/full
- **M35:** sps_cG_alex/1/full
- **M36:** sps_alex/1/full

	Percentage Accuracy (%)	
	Base	MRF
Method	Per-pixel/Per-class	Per-pixel/Per-class
M31	80.3/52.9	81.2/53.1
M32	80.7/53.9	81.9/54.4
M33	80.4/53.9	80.7/52.4
M34	79.2/48.2	80.0/48.0
M35	79.3/50.0	80.0/49.9
M36	79.0/48.6	79.4/48.6

Table 5.13: K400_S200 with baseline and best MRF on 19-class LabelMe dataset for CNN

We tested six different alternatives with $K = 400$, $S = 200$ for CNN method. Three of them are limited version and three of them are full. At the **Table 5.13**, we can see that limited version experiments are more successful than full version experiments. It seems that the best global descriptor partner is SpatialPryScaled and alex with different weights to the successful descriptors. Other alternative partners dont give the good results on per-pixel accuracy and per-class accuracy.

- **M37:** sps_alex/limited
- **M38:** sps_vgg/limited
- **M39:** cH_alex/limited

	Percentage Accuracy (%)	
	Base	MRF
Method	Per-pixel/Per-class	Per-pixel/Per-class
M37	84.3/61.9	85.5/62.8
M38	84.7/62.8	85.8/62.4
M39	84.8/60.9	86.1/62.0

Table 5.14: K200_S100 with baseline and best MRF on 19-class LabelMe dataset for CNN

We tested three different alternatives with $K = 200$, $S = 100$ for CNN method. At the **Table 5.14**, we can see that limited set of descriptors are the best experiments. It seems that the best global descriptor partner is Color Histogram and alexnet. In the CNN tests for LabelMe MRF optimization achieves some improvement on both per-pixel and per-class accuracies.

In summary for CNN models, the best result is gained when $K200\_S100$, descriptors are limited and global descriptors are Color Hist and alexnet.

	Percentage Accuracy (%)	
	Base	MRF
Method	Per-pixel/Per-class	Per-pixel/Per-class
M21	78.6/49.7	79.8/49.9
M28	78.5/49.1	79.2/48.8
M32	80.7/53.9	81.9/54.4
SuperParsing (LM) [16]	74.5/49.7	76.4/49.1
Adaptive [19]	80.3/53.3	82.7/55.1
Nguyen [6]	84.5/62.0	85.5/63.2

Table 5.15: Best results on K400_S200 for 19-class LabelMe dataset with KCB encoded

At the **Table 5.15**, we can see the best accuracy results on 19-class LabelMe dataset for $K400\_S200$. CNN method is more effective than feature encoding methods on both per-pixel and per-class accuracies. Our works are nearly same with Nguyen's Adaptive [19] which uses adaptive nonparametric approach. Other work of Nguyen [6] which uses trained object detectors, is better than ours.

At the **Table 5.16**, we can see the best accuracy results on 19-class LabelMe dataset for $K200\_S100$. Again CNN method provides really good results when

	Percentage Accuracy (%)	
	Base	MRF
Method	Per-pixel/Per-class	Per-pixel/Per-class
M24	81.3/55.6	82.9/55.5
M29	80.7/55.7	82.5/55.0
M39	84.8/60.9	86.1/62.0
SuperParsing (LM) [16]	74.5/49.7	76.4/49.1
Adaptive [19]	80.3/53.3	82.7/55.1
Nguyen [6]	84.5/62.0	85.5/63.2

Table 5.16: Best results on K200_S100 for 19-class LabelMe dataset

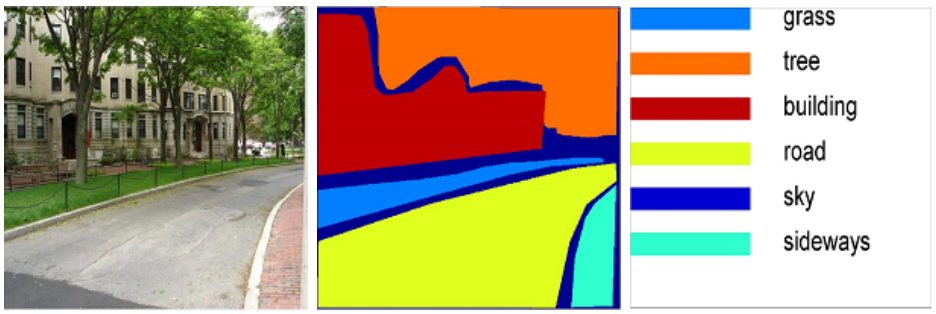
compared to feature encoding methods on both per-pixel and per-class accuracies. This time our CNN results are as good as Nguyen's [6], which are state-of-the-art in LabelMe dataset.

In summary, when using LLC encoding method, we saw 4.1% difference with baseline method on per-pixel accuracy for $K400\_S200$. When we applied for $K200\_S100$, accuracy is increasing approximately 6.8% over original SuperParsing. When using KCB encoding method, we saw 4% difference with baseline method on per-pixel accuracy for $K400\_S200$. When we applied for $K200\_S100$, accuracy is increasing approximately 6.2%. When using CNN model, we saw 6.2% difference with baseline method on per-pixel accuracy for $K400\_S200$. When we applied for $K200\_S100$, accuracy is increasing approximately 10.3%.

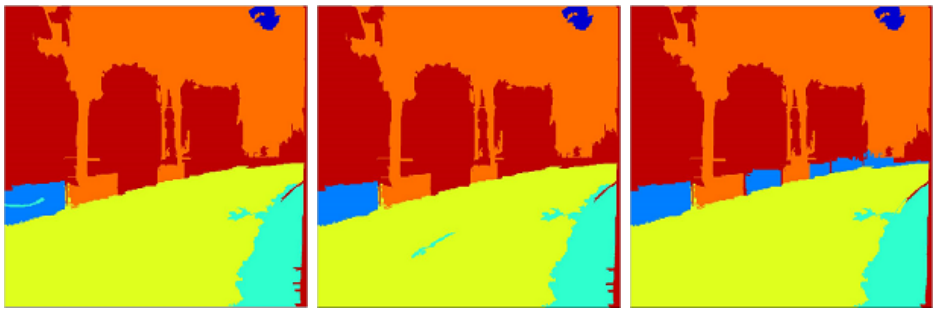
We applied three best methods to the original images. Also, visual representations give ground truth and class legends. Information of these methods as follows:

- M24:
  - LLC encoding
  - Vocabulary size: 512
  - β: 0.015
  - Distance metric: L_2
  - Superpixel color consistency (K): 200
  - Minimum superpixel size (S): 100
- M29:
  - KCB encoding
  - Vocabulary size: 256
  - σ: 100
  - Distance metric: L_1
  - Superpixel color consistency (K): 200
  - Minimum superpixel size (S): 100
- M39:
  - CNN model
  - Used global descriptors: Color Hist & alexnet
  - Weights of descriptros: 1
  - Type of segment descriptors: limited
  - Superpixel color consistency (K): 200
  - Minimum superpixel size (S): 100

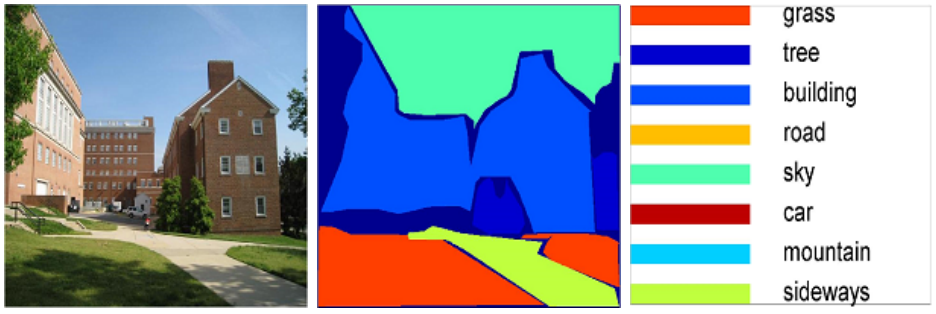
For the first original image, all methods assign road, sideways, building and trees almost correctly. However, CNN-based method assigns grass more accurately than feature encoding methods. For the second original image, all methods assign building, sky and trees almost correctly. Grass in CNN result is more accurate than in feature encoding outputs. Actually all methods assign road instead of sideways.



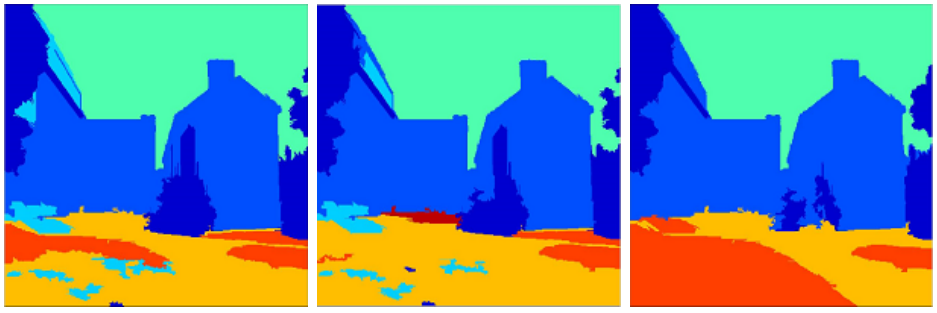
Original Image Ground Truth Legend



Method M24 Method M29 Method M39



Original Image Ground Truth Legend



Method M24 Method M29 Method M39

Figure 5.2: Visual representation of best results of 19-class LabelMe dataset

Chapter 6

Conclusion

In this project, feature encoding methods and CNN models are evaluated for superpixel image parsing. The main objective of the thesis is to show improvements in labeling accuracy percentage by using feature encoding methods as opposed to standard vector quantization method of feature vectors and by applying convolutional neural network models. LLC and KCB are the feature encoding methods which use soft quantization operation to keep more data about image. We used CNN models such as imagenet-vgg and alex-net to increase pixel accuracy. In addition to these methods and models, we added MRF optimization to obtain last labeling.

We used two different datasets to test our systems which are SIFT Flow and LabelMe datasets. We showed significant improvement on SIFT Flow dataset and LabelMe dataset over the original SuperParsing algorithm, in terms of both per-pixel and per-class accuracies.

The main contribution of this thesis is to show that feature selection is important both for global matching and superpixel matching. Better global descriptors lead to an improved retrieval set that contains training images that are more similar to the test image. An improved retrieval set also improves the superpixel matching and therefore per-pixel and per-class labeling accuracies. Advanced feature encoding methods and learned CNN features, together with standard superpixel

features, improves the accuracy of superpixel matching as well. As a conclusion, we manage to improve the labeling accuracy of SuperParsing significantly, without using any classifier training.

As a future work, we plan to extend the research in this thesis, as follows:

- The feature performances could be analyzed and optimal set of features could be determined in the trainig set. Then these features could be applied to the test set.
- Other CNN models such as GoogleNet [32], ResNet [33] can be tested.
- The CNN architectures could be re-trained with the training superpixels for learning improved features.
- Different feature encoding methods such as Fisher vectors [14] and Super-Vector Coding [15] can be used.

References

[1] H. F. Ates, S. Sunetci, and K. E. Ak, "Kernel likelihood estimation for superpixel image parsing," *ICIAR*, 2016.

[2] H. F. Ates and S. Sunetci, "Improving semantic segmentation with generalized models of local context," *CAIP*, 2017.

[3] ——, "Kernel kod-tablosu kodlamas ile sahne etiketleme," *SIU*, 2017.

[4] "Deep Learning." Wikipedia. [Online]. Available: https://en.wikipedia.org/index.php? q=aHR0cHM6Ly9lbi53aWtpcGVkaWEub3JnL3dpa2kvRGVlcF9sZWFybmluZyNEZWVwX25ldXJhbF9uZXR3b3Jrcw.

[5] L. Arnold, S. Rebecchi, S. Chevallier, and H. Paugam-Moisy, "An introduction to deep learning," *ESANN*, 2011.

[6] T. V. Nguyen, L. Liu, and K. Nguyen, "Exploiting generic multi-level convolutional neural networks for scene understanding," *ICARCV*, 2016.

[7] A. Krizhevsky, I. Sutskever, and G. Hinton, "Imagenet classification with deep convolutional neural networks," *Conference on Neural Information Processing Systems*, pp. 1106–1114, 2012.

[8] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva, "Learning deep features for scene recognition using places database," *Advances in Neural Information Processing Systems*, pp. 487–495, 2014.

[9] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and F. Li, "Large-scale video classification with convolutional neural networks," *CVPR*, pp. 1725–1732, 2014.

[10] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, pp. 650–651, 2016.

[11] K. Chatfield, V. Lempitsky, A. Vedaldi, and A. Zisserman, "The devil is in the details: An evaluation of recent feature encoding methods."

[12] J. C. Gemert, J. M. Geusebroek, V. C. J., and A. W. M. Smeulders, "Kernel codebooks for scene categorization," *Proc. ECCV*, 2008.

[13] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong, "Locality-constrained linear coding for image classification," *Proc. CVPR*, 2010.

[14] F. Perronnin, J. Sanchez, and T. Mensink, "Improving the fisher kernel for large-scale image classification," *Proc. ECCV*, 2010.

[15] X. Zhou, K. Yu, T. Zhang, and T. S. Huang, "Image classification using super-vector coding of local image descriptors," *Proc. ECCV*, 2010.

[16] J. Tighe and S. Lazebnik, "Superparsing: Scalable nonparametric image parsing with superpixels," *ECCV*, pp. 352–365, 2010.

[17] E. David and R. Fergus, "Nonparametric image parsing using adaptive neighbor sets," *CVPR*, pp. 2799–2806, 2012.

[18] M. George, "Image parsing with a wide range of classes and scene-level context," *CVPR*, pp. 3622–3630, 2015.

[19] T. V. Nguyen, C. Lu, J. Sepulveda, and S. Yan, "Adaptive nonparametric image parsing," *IEEE Trans. on Circuits and Sys. for Video Tech.*, vol. 25, no. 10, pp. 1565–1575, 2015.

[20] H. Myeong and K. M. Lee, "Tensor-based high-order semantic relation transfer for semantic scene segmentation," *CVPR*, pp. 3073–3080, 2013.

[21] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation," *Int. J. Comput. Vision 2*, 2004.

[22] J. Shotton, J. Winn, C. Rother, and A. Criminisi, "Textonboost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation," *ECCV*, pp. 1–15, 2006.

[23] A. Oliva and A. Torralba, "Modeling the shape of the scene: A holistic representation of the spatial envelope," *IJCV*, vol. 42, no. 3, pp. 145–175, 2010.

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

[25] A. Oliva and A. Torralba, "Building the gist of a scene: The role of global image features in recognition," *Visual Perception, Progress in Brain Research*, p. 155, 2006.

[26] "Convolutional Neural Network." GitHub. [Online]. Available: http://cs231n.github.io/convolutional-networks/.

[27] "Deep Learning." SlideShare. [Online]. Available: http://www.slideshare.net/datasciencekorea/5-20141107deeplearning.

[28] "Neural Network." GitHub. [Online]. Available: http://cs231n.github.io/neural-networks-1/.

[29] "Recognition Deep Learning." GitHub. [Online]. Available: https://gogul09.github.io/software/flower-recognition-deep-learning.

[30] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, "Return of the devil in the details: Delving deep into convolutional nets," *Proc. British Machine Vision Conf.*, 2014.

[31] J. Deng and W. Dong, "Imagenet: A large-scale hierarchical image database," *CVPR*, 2009.

[32] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, and S. Reed, "Going deeper with convolutions," *CVPR*, 2015.

[33] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CVPR*, 2016.

Curriculum Vitae

Sercan Sünetci was born on 22 January 1991, in İstanbul. He received his BS degree in Electrical and Electronics Engineering from Işık University in 2014 with 2^{nd} ranking out of all graduates. He worked as an project assistant in Işık University for TÜBİTAK project no: $115E307$. He is currently working as a specialist at Turkish Airlines.