Handwritten Word Spotting with Corrected Attributes
Mid-Eval Report

Team Members:

- Ritvik Agarwal (2018122005)
- Vivek Chandela (20171195)
- Kirandevraj R (2019701001)

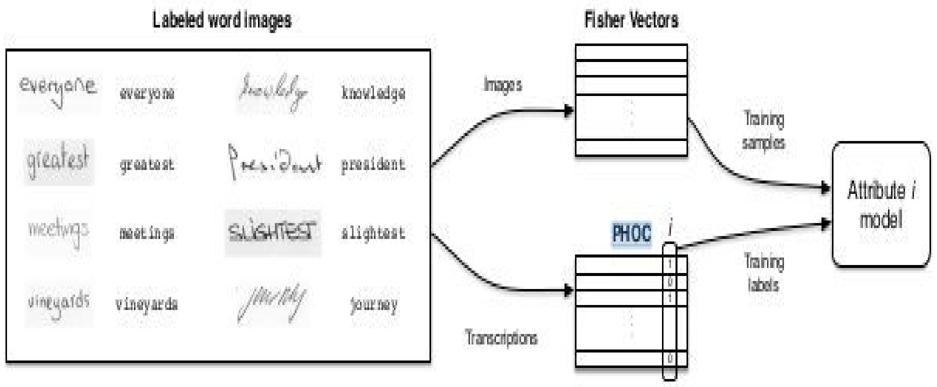
Github Repo:

https://github.com/Kirandevraj/Handwritten-Word-Spotting-with-Corrected-Attributes

Objective:

To find all instances of a given word in a potentially large dataset of document images in a multi-writer setting. The various types of Queries to be handled are:

- Query by example(Image)
- Query by string(Text)



Brief Overall Approach:

Build a unified classifier to predict the attribute representation(PHOC) from a given Fisher Vector representation computed over SIFT descriptors extracted densely from the word images. The word strings are encoded as PHOCs which are learned using SVM. After that these learned attributes and PHOCs are projected to the common subspace where they are maximally correlated. Given a QBS/QBE, get the matches with the cosine similarity score among all the document images.

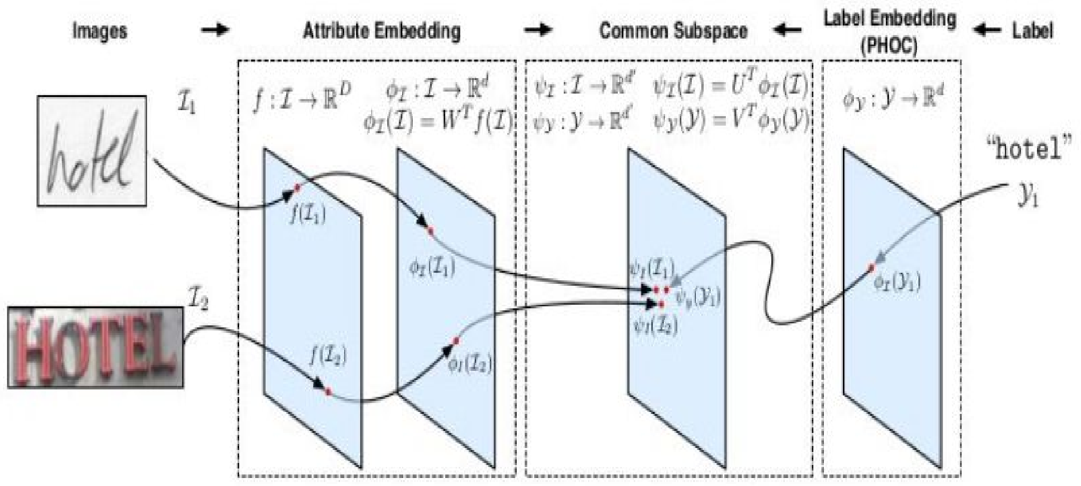

Images
Attribute Embedding
Common Subspace
Label Embedding (PHOC)
Label
$f : \mathcal{I} \to \mathbb{R}^D$
$\phi_\mathcal{I} : \mathcal{I} \to \mathbb{R}^d$
$\phi_\mathcal{I}(\mathcal{I}) = W^T f(\mathcal{I})$
$\psi_\mathcal{I} : \mathcal{I} \to \mathbb{R}^{d'}$
$\psi_\mathcal{Y} : \mathcal{Y} \to \mathbb{R}^{d'}$
$\psi_\mathcal{I}(\mathcal{I}) = U^T \phi_\mathcal{I}(\mathcal{I})$
$\psi_\mathcal{Y}(\mathcal{Y}) = V^T \phi_\mathcal{Y}(\mathcal{Y})$
$\phi_\mathcal{Y} : \mathcal{Y} \to \mathbb{R}^d$
$\mathcal{I}_1$
$\mathcal{I}_2$
"hotel"
$\mathcal{Y}_1$
$f(\mathcal{I}_1)$
$f(\mathcal{I}_2)$
$\phi_\mathcal{I}(\mathcal{I}_1)$
$\phi_\mathcal{I}(\mathcal{I}_2)$
$\psi_\mathcal{I}(\mathcal{I}_1)$
$\psi_\mathcal{I}(\mathcal{I}_2)$
$\psi_y(\mathcal{Y}_1)$
$\phi_\mathcal{I}(\mathcal{Y}_1)$

Fig1: Overview of the implemented method. Images are first projected into an attributes space with the embedding function after being encoded into a base feature representation. Then embedded labels and attributes in a learned common subspace

Timeline of Work Done:

- Extract SIFT features densely over the image using VLFeat's vl_phow implementation.
- Reduce dimension of SIFT features using PCA.
- Using 16 gaussians per GMM, train GMM using SIFT features.

- These features are then aggregated into an FV that considers the gradients with respect to the means and variances of the GMM generative model.
- For training PHOC of an image, we use levels 2, 3 and 4 as well as 75 common bigrams at level 2, leading to 384 dimensions.

1. Extracting SIFT Features:

We use the Fisher vectors as our base image representation. SIFT features are densely extracted at 6 different patch sizes (bin sizes of 2, 4, 6, 8, 10, and 12 pixels) from the images and reduced to 62 dimensions with PCA. Then, the normalized x and y coordinates are appended to the projected SIFT descriptors. To normalize the coordinates, we use the whole image as a reference system.
For calculating SIFT features, we have used VLFeat's vl_phow implementation. In this SIFT is calculated at various levels by Gaussian smoothing of the image. The resulting SIFT features calculated from vl_phow have 128 dimensions. They are reduced to 62 dimensions with the help of PCA (Principal Component Analysis). The SIFT descriptors of the image are then enriched by appending the normalized x and y coordinates and the scale they were extracted at. Thus a total of 64-dimensional vector is calculated for each SIFT.

2. Learning GMM:

To learn the GMM we use 1 million SIFT features extracted from words from the training sets. We use 16 Gaussians per GMM, and learn the GMM in a structured manner using a 2×6 grid leading to a GMM of 192 Gaussians. This produces histograms of $2 \times 64 \times 192 = 24{,}576$ dimensions. The descriptors are then power and L2 normalized.

3. Computing Fisher Vectors:

Fisher Vectors can be understood as a bag of words representation that encodes higher order statistics and has been shown to be a good encoding method.
The obtained SIFT features are hen aggregated into an FV that considers the gradients with respect to the means and variances of the GMM generative model.
Let I = (x_1 ,…, x_N) be a set of D dimensional feature vectors extracted from an image. Let Θ = (μk, Σk , πk : k = 1 ,..., K) be parameters of K Gaussians fitting the distribution of descriptors. The GMM associates each vector xi to model k in the mixture model. For each model k, consider the mean and covariance deviation vectors, where j = (1,..., D) spans the vector dimensions.

$$u_{jk} = \frac{1}{N\sqrt{\pi_k}} \sum_{i=1}^{N} q_{ik} \frac{x_{ji} - \mu_{jk}}{\sigma_{jk}},$$

$$v_{jk} = \frac{1}{N\sqrt{2\pi_k}} \sum_{i=1}^{N} q_{ik} \left[\left(\frac{x_{ji} - \mu_{jk}}{\sigma_{jk}} \right)^2 - 1 \right]$$

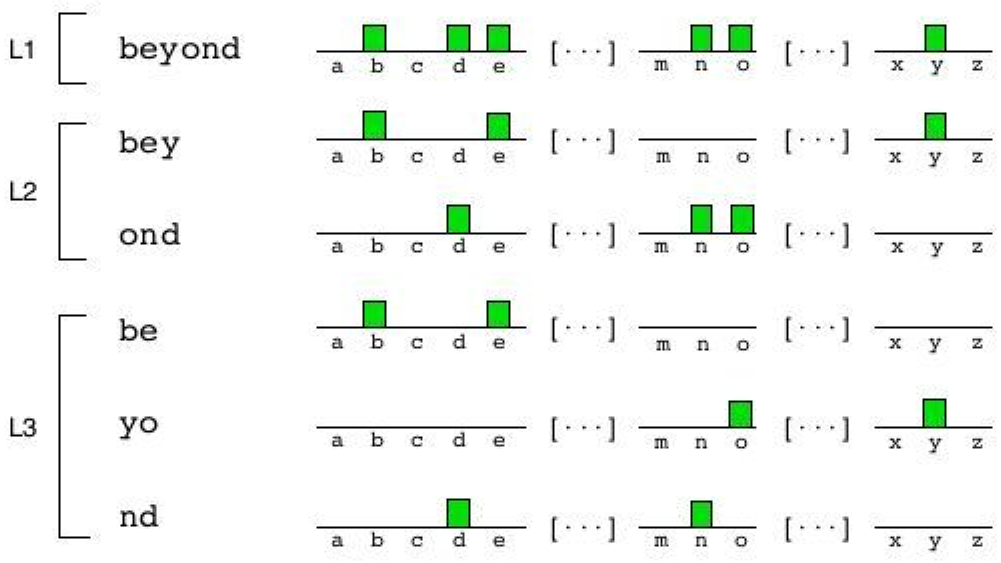
The FV of the image I is the stack of the above vectors. $\Phi(I) = [...uk...vk...]^T$
To capture the spatial features of the image, we have divided the image into 2x6 grid. Then the above procedure is applied individually to each rectangle of the grid.

Algorithm 1 Compute Fisher vector from local descriptors

```
Input:
  • Local image descriptors $X = \{x_t \in \mathbb{R}^D, t = 1, \ldots, T\}$,
  • Gaussian mixture model parameters $\lambda = \{w_k, \mu_k, \sigma_k, k = 1, \ldots, K\}$
Output:
  • normalized Fisher Vector representation $\mathscr{G}_\lambda^X \in \mathbb{R}^{K(2D+1)}$
  1. Compute statistics
      • For $k = 1, \ldots, K$ initialize accumulators
          – $S_k^0 \leftarrow 0, \quad S_k^1 \leftarrow 0, \quad S_k^2 \leftarrow 0$
      • For $t = 1, \ldots T$
          – Compute $\gamma_t(k)$
          – For $k = 1, \ldots, K$:
              ∗ $S_k^0 \leftarrow S_k^0 + \gamma_t(k)$,
              ∗ $S_k^1 \leftarrow S_k^1 + \gamma_t(k) x_t$,
              ∗ $S_k^2 \leftarrow S_k^2 + \gamma_t(k) x_t^2$
  2. Compute the Fisher vector signature
      • For $k = 1, \ldots, K$:
            $\mathscr{G}_{\alpha_k}^X = \left(S_k^0 - T w_k\right) / \sqrt{w_k}$
            $\mathscr{G}_{\mu_k}^X = \left(S_k^1 - \mu_k S_k^0\right) / (\sqrt{w_k} \sigma_k)$
            $\mathscr{G}_{\sigma_k}^X = \left(S_k^2 - 2\mu_k S_k^1 + (\mu_k^2 - \sigma_k^2) S_k^0\right) / \left(\sqrt{2 w_k} \sigma_k^2\right)$
      • Concatenate all Fisher vector components into one vector
        $\mathscr{G}_\lambda^X = \left(\mathscr{G}_{\alpha_1}^X, \ldots, \mathscr{G}_{\alpha_K}^X, \mathscr{G}_{\mu_1}^{X\,\prime}, \ldots, \mathscr{G}_{\mu_K}^{X\,\prime}, \mathscr{G}_{\sigma_1}^{X\,\prime}, \ldots, \mathscr{G}_{\sigma_K}^{X\,\prime}\right)'$
  3. Apply normalizations
      • For $i = 1, \ldots, K(2D+1)$ apply power normalization
          – $\left[\mathscr{G}_\lambda^X\right]_i \leftarrow \operatorname{sign}\left(\left[\mathscr{G}_\lambda^X\right]_i\right) \sqrt{\left|\left[\mathscr{G}_\lambda^X\right]_i\right|}$
      • Apply $\ell_2$-normalization:
        $\mathscr{G}_\lambda^X = \mathscr{G}_\lambda^X / \sqrt{\mathscr{G}_\lambda^{X\prime} \mathscr{G}_\lambda^X}$
```

4. Encoding words as PHOC:

PHOC, the pyramidal histogram of characters, is a binary histogram that encodes whether a particular character appears in the represented word or not. By using a spatial pyramid we add some coarse localization, e.g., this character appears in the first half of the word, or this character appears in the last quarter of the word (see Fig. 2). The approach embeds text strings into a d−dimensional binary space. It encodes a particular character that appears in a particular spatial region of the string. The histogram can also encode bigrams or other combinations of characters. When computing the attribute representation, we use levels 2, 3, and 4 (2 + 3 + 4)x26 = 2 34 dimensions. We also used 75 common bigrams at level 2, leading to an extra 150 dimensions for a total of 384 dimensions.



Technical Challenges:

- Very fine-grained classes - difference of one character is a negative result.
- Very high intra-class variability - different writers may have very different writing styles.
- Variable-length features are unsuitable due to 2 reasons:

- OOV (out of vocabulary) not possible so, only a known, limited number of keywords can be used as queries.
- Computing distance between words is very slow at test time.
- Although they are more flexible than fixed-length sequences, they don’t leverage supervised info to learn similarities and differences between different writing styles.

Implementation Challenges:

- Lack of resources related to the implementation details of the original research paper.
- High Computational power required as the dataset is huge (~1M images). Hence, rented google cloud’s virtual machine.
- Since we’re using Dense SIFT many features were coming out to be zero which was resulting in erroneous results. Excluded such features.
- Initially used sklearn’s GMM but it had convergence issues. So used vlfeat’s implementation of GMM.
- Used vlfeat’s python implementation of FV as our own implementation in python was slow.

Future Work:

- Predict/train the PHOC attributes using a SVM classifier, given the FV.
- Since we have the image and string for a word, actual PHOC attributes can be found by using the string input
- Using CCA (Canonical Correlation Analysis), get the projections of the predicted scores and the ground truth values
- Use cosine similarity to compute the mean average precision.

References:

1. http://www.cvc.uab.es/~almazan/wp-content/uploads/2013/10/almazan_ICCV13.pdf
2. http://www.cvc.uab.es/~afornes/publi/journals/2014_PAMI_Almazan.pdf
3. http://www.vlfeat.org/
4. https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.SGDClassifier.html
5. https://scikit-learn.org/stable/modules/multiclass.htm