Word Spotter

Praveen Balireddy (2018201052)
Aman Joshi (2018201097)
Abhijeet Panda (2018201044)

Objective

Find all instances of a given word in a potentially large dataset of document images.

Queries:

1) Query by example(Image)
2) Query by string(Text)

Challenges with previous works:

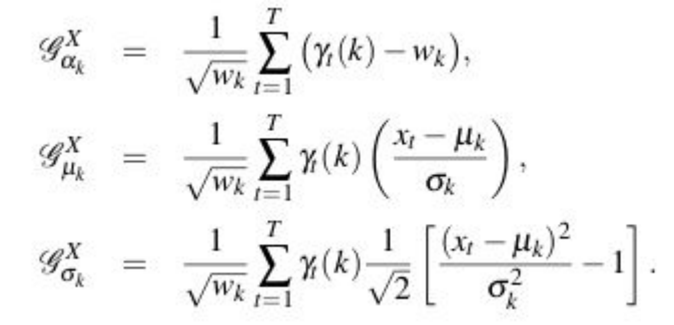
1) Out of Vocabulary words(words not there in training data, but exist in test)
2) Time taken for the image retrieval
3) Same word, different handwritings

Current approach:

1) Instead of learning models for particular keywords, learning what makes words and letters unique independently of their writers' style.
2) Using an attribute based representation for each word.

Fisher Vector

1) A Gaussian Mixture Model (GMM) is used to model the distribution of features (e.g. SIFT) extracted all over the image
2) The Fisher Vector (FV) encodes the gradients of the log-likelihood of the features under the GMM, with respect to the GMM parameters.



$$
\begin{aligned}
\mathcal{G}^X_{\alpha_k} &= \frac{1}{\sqrt{w_k}} \sum_{t=1}^{T} \left(\gamma_t(k) - w_k\right), \\
\mathcal{G}^X_{\mu_k} &= \frac{1}{\sqrt{w_k}} \sum_{t=1}^{T} \gamma_t(k) \left(\frac{x_t - \mu_k}{\sigma_k}\right), \\
\mathcal{G}^X_{\sigma_k} &= \frac{1}{\sqrt{w_k}} \sum_{t=1}^{T} \gamma_t(k) \frac{1}{\sqrt{2}} \left[\frac{(x_t - \mu_k)^2}{\sigma_k^2} - 1\right].
\end{aligned}
$$

PHOC(pyramidal histogram of characters

1) This binary histogram encodes whether a particular character appears in the represented word or not.
2) Spatial pyramid representation ensures that the information of the characters order is preserved

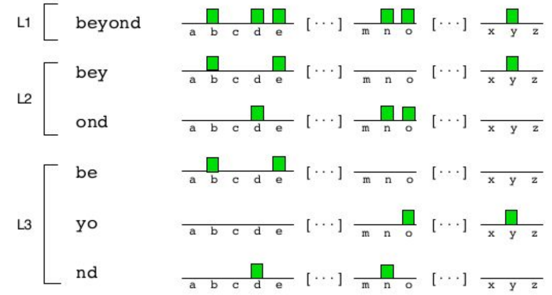


Figure 1. PHOC histogram at levels 1, 2, and 3. The final PHOC histogram is the concatenation of these partial histograms.

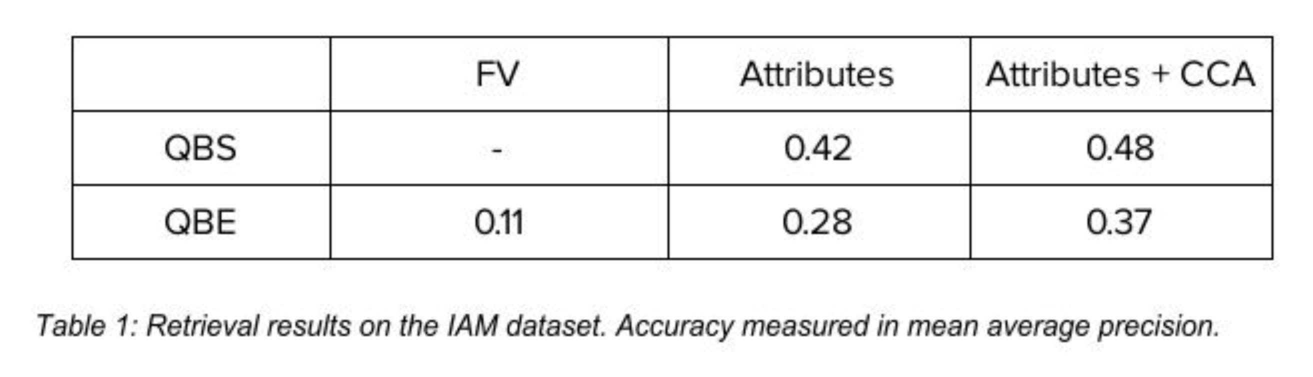
Algorithm:

1) SIFT features are densely extracted from the images over a 2x6 spatial grid and reduced to 62 dimensions with PCA
2) Normalized x and y coordinates are appended to the projected SIFT descriptors
3) Predict/train the PHOC attributes using a SVM classifier, given the FV
4) Since we have the image and string for a word, actual PHOC attributes can be found by using the string input
5) Using CCA (Canonical Correlation Analysis), get the projections of the predicted scores and the ground truth values
6) Use cosine similarity to compute the mean average precision

Implementation Details:

1) Used 1 million SIFT features over 2 x 6 spatial grid for training the GMM with 16 gaussians .
2) Used PCA to reduce dimensions to 64. This produces histograms of 2 × 64 × 192 = 24, 576 dimensions. The descriptors are then power and L2-normalized.
3) When computing the attribute representation, we used levels 2, 3 and 4 as well as 75 common bigrams at level 2, leading to 384 dimensions considering the 26 characters of the English alphabet.
4) For learning the attributes we've used 39756 (40%) images to train one vs rest SGD classifier.
5) For CCA we've used 41032 images to learn the common subspace. We've reduced the 384 dimensions to 196 dimensions in the process.

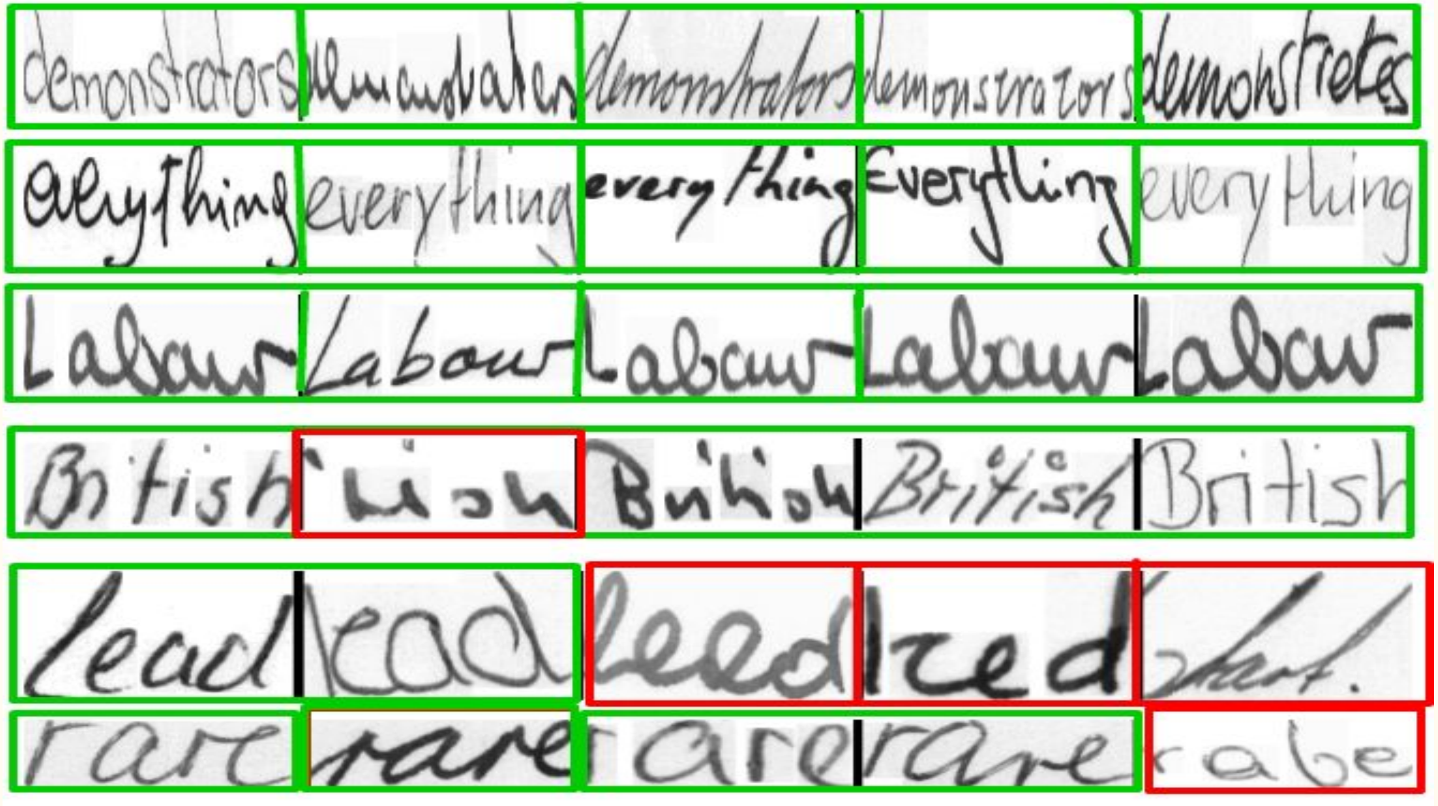
Results:



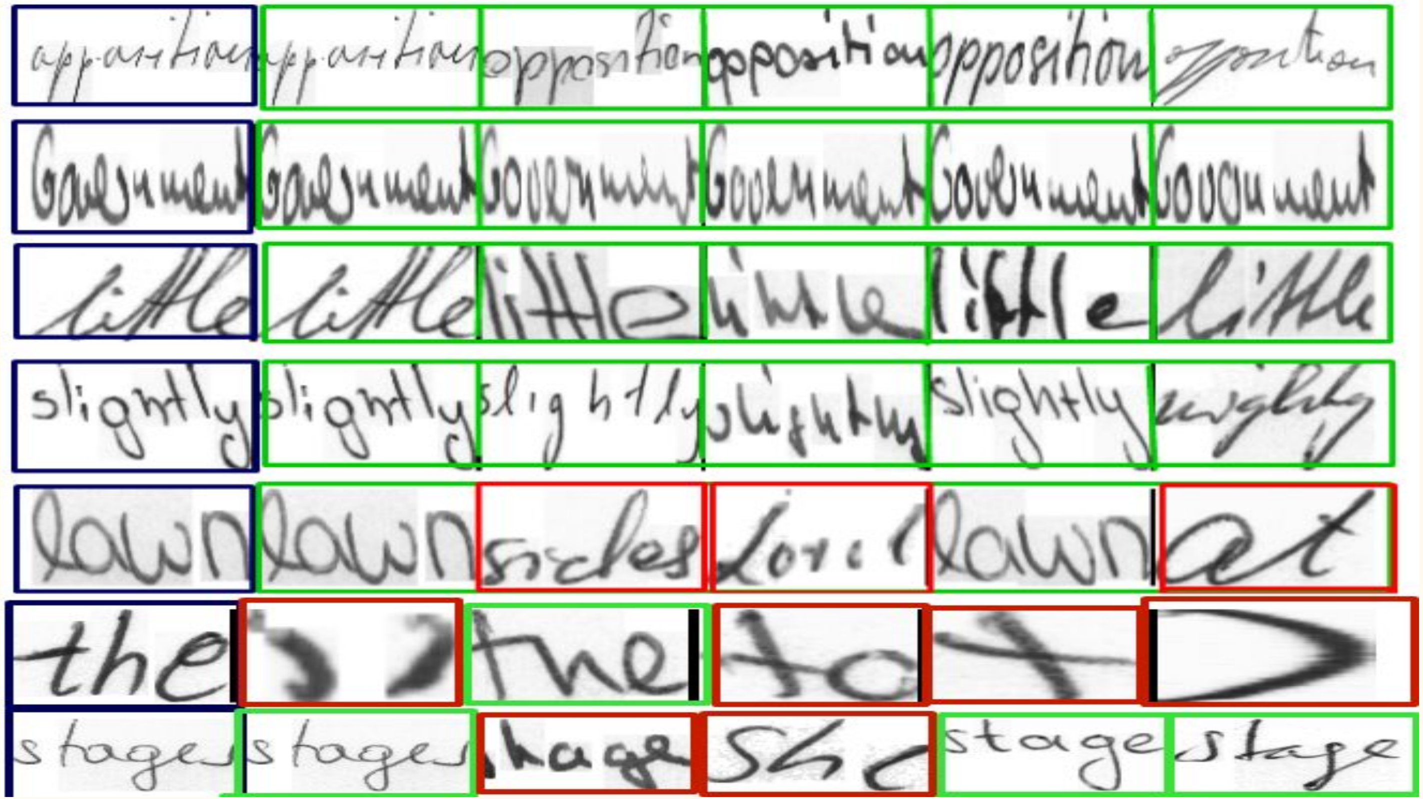
	FV	Attributes	Attributes + CCA
QBS	-	0.42	0.48
QBE	0.11	0.28	0.37

Table 1: Retrieval results on the IAM dataset. Accuracy measured in mean average precision.

QBS:


demonstrators demonstrators demonstrators demonstrators demonstrates
everything everything everything Everything everything
Labour Labour Labour Labour Labour
British British British British
lead lead leed
rare rare rare rare rabe

QBE:



Observations:

1) The MAP score with just the FV as attribute representation is used as a benchmark score for other embedding techniques.
2) The attribute based representation gives a better result than the benchmark as the SVM learns the handwriting differences and what factors that make a word unique.
3) Initially when λ = 1e-3 the SVM was overfitting hence was giving poor results. Later for λ = 1e-5, the model was performing better.
4) Also there are some mismatches in the resulting images above, this is happening as the SVM might be overfitting and not have seen similar images in training.

Challenges:

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

Thank You!