Medical image retrieval based on 3D lesion content

Blaine Rister

December 11, 2015

Abstract

Content-based image retrieval is an emerging technology which could provide decision support to radiologists. This paper describes a system for content-based image retrieval based on 3D features extracted from liver lesions in abdominal computed tomography images. A supervised learning algorithm is developed to transform image features into search rankings. In our experiments, the supervised learning provides some benefit over unsupervised methods.

1. Introduction

Content-based image retrieval is the task of searching a database for similar images to a given query image. Rather than relying on metadata, as do many commercial search engines, content-based retrieval systems use image processing and computer vision to describe the content of an image. For example, two images having similar color distributions might be marked as similar. Content-based retrieval algorithms can augment metadata-based retrieval, or replace it in application domains for which metadata is unavailable or unreliable.

This report describes a content-based image retrieval system for 3D medical images. Rather than searching by holistic image features, which are irrelevant in the medical context, the system extracts features directly from lesions, ranking each lesion on a case-by-case basis. Lesion similarity is predicted by a supervised learning algorithm using 3D image features as input.

Such a system could benefit practicing radiologists. Some diseases are very rare, such that a radiologist might only see a handful of cases in his or her career. In these situations, a radiologist could search through a database for similar cases to gain more insight into the problem.

2. Background and prior work

Content-based image retrieval is a well-established research area, albeit less so in the medical context. The reader can refer to Akgul et al. for a survey of the field and discussion of challenges faced in developing CBIR systems for medical use (Akgul et al., 2011). This project is a follow-up on the work of Napel et al., who studied content-based image retrieval using the same image dataset, but with different features and learning algorithms (Napel et al., 2010). Napel et al. used a combination of image features based on shape and texture, along with a boosting-like learning algorithm, to predict pairwise similarity between lesions. Hoffmaniger et al mined radiology text reports to extract training supervision from a larger dataset. Despite previous attempts at medical content-based image retrieval, there is still room for improvement on the state-of-the-art.

This project differs from previous work in two aspects. Firstly, it uses recently-developed 3D image features, constituting an improvement over 2D features and some previous attempts at 3D features. Secondly, it uses a modified regression algorithm that specifically takes the ranking problem into account.

3. Dataset and features

Before describing the details of our training data, we will brush up on the basics of computed tomography (CT) scans. CT scans are essentially three-dimensional X-ray images, reconstructed from multiple one-dimensional measurements taken by a moving measurement device. CT scans are among the most popular types of medical images, with over 72 million performed in the United States alone in the year 2007 (de Gonzalez A et al., 2009). They are commonly used to diagnose and monitor cancer, among other diseases. As seen in figure 1, liver lesions are clearly visible in CT scans as dark areas surrounded by the lighter liver tissue.

Final project report for CS 229 at Stanford University, 2015.

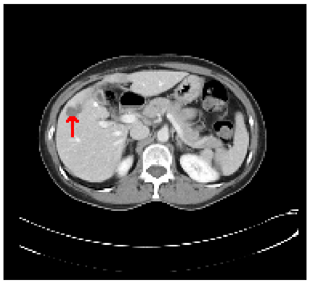

Figure 1. Example liver lesion, indicated by the red arrow, in a CT scan.

Our dataset consists of 30 annotated liver lesions from CT scans. The annotations provide a few points on the boundary of each lesion, and were created by board-certified radiologists. These annotations allow us to extract features only from the lesions, disregarding other image content.

Each CT scan contains a massive amount of data, commonly having an (x, y, z) resolution of around $0.7 \times 0.7 \times 1.2$mm. This is significantly reduced by extracting the lesions, which may have a radius of around 20 voxels. Still, given our limited dataset, it is not prudent to train on image data directly. Rather, we extract a set of image features describing the shape and texture of each lesion, allowing us to summarize the relevant content in a low-dimensional space. The following sections describe the two types of features used for training.

3.1. SIFT3D features

The first type of features are 3D image descriptors, previously designed for a different work. Here I will give a rough overview of their properties and motivation, without delving into too much detail, as image feature engineering is outside the scope of CS 229. The features are a higher-dimensional analogue of the Scale-Invariant Feature Transform from the computer vision literature (Lowe, 2004).

The input to the feature extraction program is a set of abdominal computed tomography (CT) scans, and a set of annotations of the liver lesions in those scans. The annotations are ordered sets of points tracing the boundary of each lesion. We assign a coordinate $\boldsymbol{x} = (x, y, z)$ and scale σ to each lesion by taking the center and radius of the minimum bounding sphere of the annotation points. Assuming the annotations scale consistently with the images, the scale parameter ensures scale-invariance of the features.

It remains to extract the feature vectors. There are many variants on this idea, but the approach used here is an array of weighted histograms describing the distribution of image gradient orientations. That is, for each pixel in some window $W \ni \boldsymbol{x}$, we approximate the gradient of the image data $\nabla I_{\boldsymbol{x}}$, and add this vector to a weighted histogram based on its orientation, as shown in figure 2. In practice, this is a robust representation of the shape of an object about a coordinate $\boldsymbol{x}$.

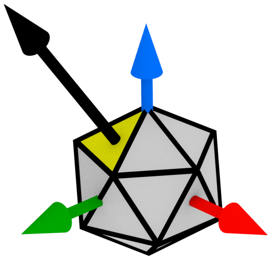

Figure 2. A gradient vector, shown in black, intersecting a histogram tile, shown in yellow.

3.2. Haralick texture features

The second type of features are Haralick texture features. These were originally developed by Haralick et al. for classification of satellite images (Haralick et al., 1973). Haralick used his simple texture features, together with piecewise linear regression, to accurately classify land into several usage categories, such as city and farmland.

Haralick's texture features are computed from a matrix P, which is known as the gray-level co-occurence matrix (GLCM). Its elements P_{ij} give an estimate of the probability that a pixel of intensity i neighbors one of intensity j. Note that this joint probability distribution is discrete, which assumes that the image intensity values are quantized. This is not terribly inaccurate, as images are captured and stored in a quantized form. Knowing this, it is straightforward to estimate these probabilities as

$$P_{ij} = \frac{\#\text{ pixels } i \text{ neighboring pixels } j}{\text{total } \#\text{ pixels}}.$$

Note that this matrix is computed only within the lesion boundaries, as we are trying to describe only the texture of the lesion itself. Here we have used an informal notation, as the concept of one pixel neighboring another is quite easy to intuit, but less so to formalize algebraically.

For our purposes, not enough to use the matrix P as training input. Images are commonly quantized into 256 different intensity values, in which case P is of size

Contrast	Correlation	Energy	Homogeneity
.36	.92	.46	.97

Table 1. Texture features for the lesion from figure 1.

256×256. This large matrix could hardly be called a summary of the image data. Thus, Haralick computed a feature vector of statistics summarizing the content of P:

$$\begin{aligned}
\text{Contrast} &= \sum_{ij} P_{ij} |i-j|^2 \\
\text{Correlation} &= \sum_{ij} P_{ij} \frac{(i-\mu_i)(j-\mu_j)}{\sigma_i \sigma_j} \\
\text{Energy} &= \sum_{ij} P_{ij}^2 \\
\text{Homogeneity} &= \sum_{ij} \frac{P_{ij}}{1+|i-j|}
\end{aligned}$$

where $\mu_i, \sigma_i, \mu_j, \sigma_j$ are the mean and standard deviation of the marginal distributions P_i, P_j, respectively. Note that Haralick originally defined 13 different texture features, but we list only those used in this work. We refer the reader to the original article for the interpretation of these features (Haralick et al., 1973).

As an example, table 1 shows the four texture features for the lesion from figure 1.

4. Learning algorithm

Having described the dataset and features, we will now formally express the goals of the CBIR system, and derive a machine learning algorithm approximating those goals. Section 4.1 describes the ranking problem and quantification of ranking accuracy, while section 4.2 describes a regression algorithm for predicting the relevance scores.

4.1. Ranking overview

Having extracted image features, it remains to use machine learning to assign search rankings to a query point. Given a query feature vector, we must generate a permutation of the vector $(1, 2, ..., m)$ ranking the m training features in terms of relevance to the query, where 1 is the most relevant and m the least.

To this end, we derive the following supervised learning algorithm: given a query feature x, predict the relevance to each training feature $x^{(i)}$. Then, compute the search ranking function $r_j(x)$ by sorting the relevance scores. In the literature, this is known as the *point-wise* approach to the ranking problem, as it considers only the relationship between the query image and an individual training data point. In total, this yields m distinct regression problems, each with m training examples $(x^{(j)}, A_{ij})$.

The sorting step is justified upon considering the goal of the learning algorithm. Our objective is to maximize the normalized discounted cumulative gain (NDCG) from cross-validation. NDCG is a widely-used measure of search performance. First, define the discounted cumulative gain (DCG) of the j^{th} query as

$$\text{DCG}_j = \sum_{i=1}^{p} \frac{2^{y_i} - 1}{\log_2(i+1)}$$

where $y_i = A_{r_j(i),j}$ is the relevance score of the i^{th} result in the j^{th} query.

Having defined the DCG, the NDCG is simply

$$\text{NDCG}_j = \frac{\text{DCG}_j}{\text{DCG}_j^*},$$

where DCG_j^* is given by the optimal ranking function for the j^{th} query. From here, it is clear that, given a set of predicted relevance scores, sorting the scores yields the optimal NDCG.

4.2. Regression for optimal ranking

In the previous section, we described a machine learning approach to ranking using m distinct regression problems. In this section, we will formalize the regression problem in an attempt to achieve optimal ranking. We will see that, for most datasets, optimal ranking is not possible within our framework, and instead strike a compromise between data fidelity and ranking sub-optimality.

4.2.1. Pairwise distance features

Before delving into the specifics of the ranking problem, we note a useful transformation of the image features specific to this application. Given a query image, we wish to predict its similarity to a training image. Thus, it is not the image features themselves that are of interest, but the differences between them. We therefore define a new feature vector as follows: let $q_s, t_s \in \mathbb{R}^{768}$ be the SIFT3D features from the query lesion q and the training lesion t, respectively. Similarly, let $q_{t1}, ..., t_{t4}, q_{t1}, ..., ty$ be the four texture features from the query and training images, respectively. Then, the pairwise distance feature vector x is computed as

$$x = \left(\|q_s - t_s\|_2, \quad |q_{t1} - t_{t2}|, \quad ..., \quad |q_{t4} - t_{t4}| \right).$$

Note that this is a nonlinear feature mapping from the 772-dimensional image feature space to a 5-dimensional space. This mitigates overfitting on our small dataset. Furthermore, we have observed no performance loss from this mapping.

4.2.2. Optimal Ranking

We wish to find a mapping $\phi : \mathbb{R}^n \to \mathbb{R}$ such that $\phi(x^{(i)})$ gives an optimal ranking. Rankings are obtained by ordering $\phi(x^{(i)})$, such that $x^{(i)}$ is ranked prior to $x^{(j)}$ if and only if $\phi(x^{(i)}) > \phi(x^{(j)})$. Let $x^{(1)}, y^{(1)}$ denote the most relevant item, $x^{(2)}, y^{(2)}$ the second-most, etc. In this notation, x is the pairwise feature vector from section 4.2.1, and y is its corresponding relevance score with respect to the training image. Any solution satisfying the following constraints will produce an optimal ranking:

$$\begin{aligned} \phi(x^{(1)}) &> \phi(x^{(2)}) \\ &... \\ \phi(x^{(N-1)}) &> \phi(x^{(N)}). \end{aligned}$$

Converting to standard form, we have

$$\phi(x^{(i-1)}) - \phi(x^{(i)}) < 0.$$

This is a convex set if $\phi(x^{(i)}) - \phi(x^{(j)})$ is a convex function of the parameters of ϕ. In the case that $\phi(x) = a^T x + b$, we have

$$\begin{aligned} a^T x^{(i-1)} - a^T x^{(i)} &< 0 \\ a^T (x^{(i-1)} - x^{(i)}) &< 0 \end{aligned}$$

which is clearly convex.

4.2.3. Sub-optimal compromise

We have shown in section 4.2.2 that the set of affine regressors resulting in an optimal ranking of the training set is convex. Thus, we can quickly find a regressor which optimally ranks the training set, if one exists. However, this is not so useful in practice, as for most datasets with more training examples than feature dimensions, there is no regressor satisfying these equations, i.e. the set of optimal regressors is empty. To avoid this difficulty, we must strike a compromise by allowing some slack in the constraints. In this case, we would like to differentiate between sub-optimal solutions according to their fidelity to the training relevance scores. Thus, we arrive at the following convex optimization problem:

$$\begin{array}{ll} \text{minimize} & \frac{1}{N}\sum_{i=1}^{N} \left(a^T x^{(i)} + b - y^{(i)}\right)^2 + \alpha \|a - z\|_2 \\ \text{subject to} & z^T(x^{(2)} - x^{(1)}) < \beta \\ & ... \\ & z^T(x^{(N)} - x^{(N-1)}) < \beta \end{array} \qquad (1)$$

where N is the number of training examples, z is a vector of dummy variables, and α, β are constant parameters.

Let us dissect this algorithm. The first term of the objective is just the mean squared error of the predicted relevance scores. The dummy variables z are constrained to lie within the set of affine regressors giving no more than a β-suboptimal ranking. The term$\|a - z\|_2$ gives the distance between the regressor a and this β-suboptimal set. The parameter α negotiates between the often-competing objectives of data fidelity and optimal ranking.

This seemingly complicated model can be simplified. In practice, we replace $<$ with $\leq$, as most solvers do not differentiate between the two. In this case, if $\beta = 0$, the only feasible solution is often $z = \mathbf{0}$. Substituting this into the objective, we have

$$\text{minimize} \frac{1}{N}\sum_{i=1}^{N} \left(a^T x^{(i)} + b - y^{(i)}\right)^2 + \alpha \|a\|_2 \qquad (2)$$

which is just regularized linear regression. In fact, we shall see in section 5 that our algorithm with reasonably small β performs essentially the same as regularized linear regression. Note, however, that with $\beta > 0$ the optimal value of z is not necessarily $z = \mathbf{0}$, so the algorithm is not identical to regularized linear regression.

5. Results

We computed the performance for a variety of subsamples of the training set, using leave-one-out cross-validation. The cross-validation scheme is as follows: for each round, we withhold one lesion from the dataset. Then, we train all $N-1$ regressors predicting the relevance of a query image to each of the $N-1$ remaining training lesions. This is done without the benefit of the withheld lesion. Finally, we predict and rank the similarities of the withheld lesion to each of the $N-1$ training lesions. This gives the ranking results from a single query, from which we compute a single NDCG. Repeat this process N times, each time withholding a different lesion. The final performance estimate is the average NDCG over all N folds.

The parameters for our algorithm from problem 1 were $\alpha = 0.1, \beta = 1$. We solved the optimization problem with CVX, a popular modeling tool for convex programs (Grant & Boyd, 2014). For comparison, we computed three additional scores. The first is the worst possible NDCG for this data, averaged over all folds, computed from the worst possible ranking. The second is the score from the regularized linear regres-

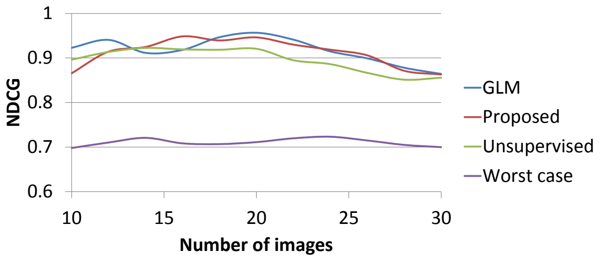

Figure 3. Mean NDCG for the various algorithms. Possible scores range between the purple line and 1.

sion problem 2, which is trained by the GLMnet library, with cross-validation for the regularization parameter α (Glm). This shows the close relationship between regularized linear regression and our algorithm. The third is an unsupervised algorithm, showing the benefit of supervised learning. The unsupervised ranking algorithm simply takes $y = \exp(-\|x\|_2)$, where $\|x\|_2$ estimates the similarity of the query and training lesion, and e^{-x} is a positive, monotonically decreasing function. Results for all of these algorithms are shown in figure 3.

In most applications, figure 3 would be called a "learning curve," hopefully showing the benefit of an increased training set. However, for each round we rank all N training items. The ranking problem becomes more difficult as we have more items to rank, and more opportunities to rank a highly-relevant item far down the list. Thus, we can see that the NDCG increases initially, but past the midpoint of the graph the benefit of more training data is outweighed by the increased difficulty of the task.

The results show that supervised learning outperforms unsupervised, although all three results are encouraging. Our algorithm performs almost identically with regularized linear regression, a result theoretically explained in section 4.2.3. Although they are not technically the same for $\beta > 0$, this investigation could be used as a theoretical justification for the use of regularization in the ranking problem. Despite encouraging results, we believe better performance is necessary for clinical usefulness. Possible improvements will be discussed in section 6.

6. Conclusions and future work

We have extracted image features from a dataset of CT liver lesions, derived a supervised learning algorithm for ranking, and reported the results in a cross-validation framework. Results were encouraging, but leave room for improvement.

In the future, we plan to extend the algorithmic ideas from section 4.2. We might try adding features, or mapping features to a higher-dimensional space, in an attempt to produce a less-trivial optimal-ranking set. Almost paradoxically, optimal ranking is guaranteed only when the number of parameters equals or exceeds the number of training elements, a cardinal sin of machine learning. The problem might be alleviated by use of a more complicated regressor, but it is challenging to find one for which the problem remains convex.

Another possible avenue of exploration is optimizing over all regressors simultaneously. In this work we have penalized the suboptimality of the ranking of a single regressor. But since the outputs of different regressors are directly compared in a search query, should not we enforce optimal ranking across regressors as well? We might also enforce symmetry between regressors, i.e. $a_i^T x_j + b_i = a_j^T x_i + b_j$. There is no shortage of avenues for exploration, but it is not clear which are both nontrivial, i.e. more substantial than mere regularization, and computationally tractable.

References

Glmnet. Available at http://web.stanford.edu/~hastie/glmnet_matlab/, accessed November 13th, 2015.

Akgul, Ceyhun Burak, Rubin, Daniel L., Napel, Sandy, Beaulieu, Christopher F., Greenspan, Hayit, and Acar, Burak. Content-based image retrieval in radiology: Current status and future directions. *Journal of Digital Imaging*, 24(2):208–222, 2011.

de Gonzalez A, Berrington, M, Mahesh, K, Kim, and et al. Projected cancer risks from computed tomographic scans performed in the united states in 2007. *Archives of Internal Medicine*, 169(22):2071–2077, 2009. doi: 10.1001/archinternmed.2009.440.

Grant, Michael and Boyd, Stephen. CVX: Matlab software for disciplined convex programming, version 2.1. `http://cvxr.com/cvx`, 2014.

Haralick, R.M., Shanmugam, K., and Dinstein, Its'Hak. Textural features for image classification. *Systems, Man and Cybernetics, IEEE Transactions on*, SMC-3(6):610–621, Nov 1973. ISSN 0018-9472. doi: 10.1109/TSMC.1973.4309314.

Lowe, David G. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, November 2004. ISSN 0920-5691. doi: 10.1023/B:VISI.0000029664.99615.94. URL `http://dx.doi.org/10.1023/B:VISI.0000029664.99615.94`.

Napel, Sandy A., Beaulieu, Chistopher F., Rodriguez, Cesar, Cui, Jingyu, Xu, JiaJing, Gupta, Ankit, Korenblum, Daniel, Greenspan, Hayit, Ma, Yongjun, and Rubin, Daniel L. Automated retrieval of ct images of liver lesions on the basis of image similarity: Method and preliminary results. *Radiology*, 256(1), July 2010.