Exploiting Image Similarity to Perform Machine Translation

Lauren Tran

March 08, 2013

Abstract

Although many distinct languages exist across cultures, one mode of communication is universal. We can exploit the universal nature of visual images in order to conduct automatic machine translation, by using Google image results across languages. With a multilingual set of images, we begin with the bag-of-visual words model, where we extract salient features through the Scale-Invariant Feature Transform. We then learn a visual vocabulary by using k-means clustering to group these features into visual words, and we define each training image in terms of our vocabulary. Using our feature vectors generated through quantizing our images, we perform an image-to-image comparison using two similarity measures: cosine similarity and the Spearman correlation coefficient. Our similarity measures allow us to construct a novel feature vector that takes advantage of global information. A final comparison of our resulting global vectors, measured by calculating euclidean distances, gives us performance that surpasses the state-of-the-art method.

1 Introduction

The standard approach to machine translation (MT) uses distributional semantics and statistics over large parallel text corpora. Parallel data, however, is not always available. By using universal information found in images, we can eliminate the need for parallel data, giving us a strong motivation to conduct automatic MT with visual information. With a lexicon of 500 English words, we conduct translations from English to five other languages using weakly-labelled web images. These images were collected by Bergsma and Van Durme [1] through automatic Google thumbnail scraping, made publicly available online. The procedure for our approach includes the following steps:

1. Extract salient features with Lowe's Scale-Invariant Feature Transform (SIFT) [2].
2. Define a vocabulary of visual words using k-means clustering.
3. Construct a histogram showing the visual word frequencies for each training image.
4. Conduct an image-to-image comparison using the cosine similarity and Spearman correlation measures.
5. Construct a global feature vector for each word that represents the one-to-all similarity of the against every word in the lexicon.
6. Identify matches by computing the distance between all words in English

This paper presents the methods and approaches we used to perform our automatic MT task in section 2, followed by the results of initial experiments in section 3, representing performance on English to Spanish words. In section 4, we present a variety of extensions and next steps for our project, along with concluding remarks. We also include an additional section with other notes and remarks regarding the progression of this project.

2 Methods

Beginning with the bag-of-visual-words model, we follow the pipeline of extracting features, defining a vocabulary of visual words, and quantizing our images. Once we have our histograms to use as preliminary feature vectors, we deviate from the traditional bag-of-visual-words model. We perform an image-to-image comparison to generate similarity scores between every English class, E, and foreign language class, F. Using our scores, we construct a final feature vector that includes global information by creating a one-to-all representation of class E to all classes F.

2.1 Feature Extraction

With the image data sets from [1], we gather information from the images that we can use to identify matches. Using SIFT [2], we extract salient features from each of the images. The SIFT detector locates features that are scale and rotation invariant. The features are also robust against affine transformations, as well as changes in noise level, 3D viewpoint, and illumination. The SIFT keypoint descriptor provides a distinctive representation of each feature. These properties prove fundamental in our application, because instances of each class can vary significantly. We plan to experiment with extracting meta-class features, developed by Bergamo and Torresani [3], which are shown to be very effective in improving accuracy results.

2.2 Defining a Visual Vocabulary

We use the features we have extracted to define a visual vocabulary, which we can later use to meaningfully describe our training images. After using SIFT to retrieve keypoints and compute descriptors, we apply k-means clustering to cluster the feature vectors into visual words. This clustering method allows us to segment our large collection of features into similar groups.

The k-means algorithm is an iterative method that partitions data by assigning a nearby cluster center to each data element. The method begins by initializing k cluster centers by randomly selecting k vectors from the set of feature vectors. It then creates k partitions by assigning each vector to the nearest center. The goal is to minimize the sum of squared differences between the cluster centers and in-cluster vectors. The algorithm iteratively assigns each feature to the nearest cluster center and updates each center to the mean of its constituents until the sum of squared differences cannot be further minimized. The algorithm converges when no vector can be reassigned.

K-means clustering outputs k clusters, each of which represents a visual word. Figure 1 shows a sample output, where k = 2. Each blue point has been assigned to the first center and each red point to the other. In this example, only two clusters are generated, as the input k value is 2. Thus far, we have experimented with two values of k — 1000 and 10000 — which we selected based on results from Bergsma and Van Durme.

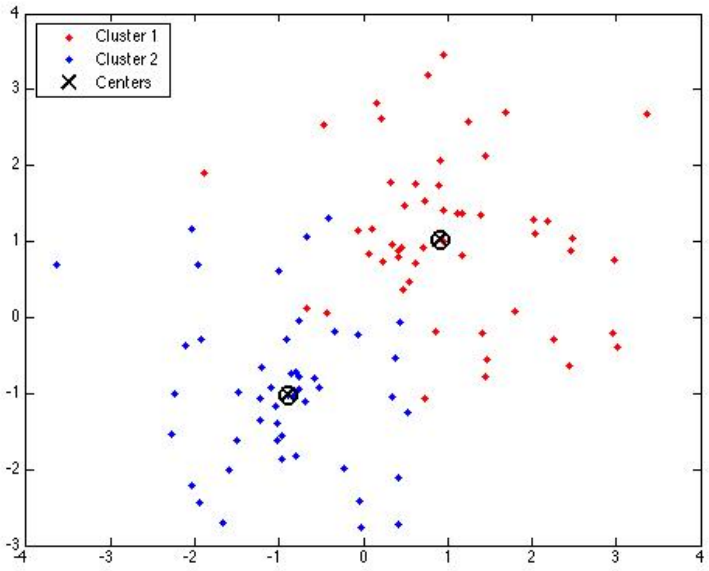

Figure 1: K-means clustering output where k = 2

.

2.3 Representing Images as Feature Vectors

To represent our images, we generate a histogram of visual word frequency for each image by counting the number of times each visual word from our vocabulary appears in the image. These histograms are our preliminary feature vectors. Next, we calculate the similarity between images in each class E to all classes F. To do so, we follow the approach presented in [1], as shown in equation 1:

$$AvgMax(E,F) = \frac{1}{|E|}\sum_{e \in E} \max_{f \in F}(cosine(e,f)) \tag{1}$$

The average maximum finds, for each $e \in E$, the best matching image $f \in F$ according to the cosine similarity measure. We experimented with additional similarity measure and found the Spearman correlation to significantly improve results, as it measures correlation as opposed to distance. We find the cosine similarity between images e and f by calculating

$$cosine(e,f) = \frac{e * f}{|e||f|} \tag{2}$$

and the Spearman correlation by evaluating

$$Spearman(e,f) = \frac{6\sum_{i=1}^{k}(e_i - f_i)^2}{k(k^2-1)}, \tag{3}$$

where k is the number of histogram bins, as defined in our k-means clustering step.

With our avgmax scores, we can construct our global feature vector for each class, as follows. For each E, we create a vector containing it's similarity score against every E, followed by its score against every F. We do the same for each F, in order to create alignment in our feature vectors. Figure 2 illustrates the construction of our global vector, for English word *cat* and Spanish word *gato*. In this case, the English lexicon contains *cat*, *tree*, and *dog*, while the Spanish lexicon contains *gato*, *arbol*, and *perro*. Note that the order of the words matters within a lexicon but not across lexicons. For instance, we must compare *cat* and *gato* to each English word in the same order — in this case, we have $\{cat, tree, dog\}$. We may, however, change the order when comparing *cat* and *gato* to each Spanish word — here, we do not need to compare to $\{gato, arbol, perro\}$ but may instead use $\{arbol, perro, gato\}$. This reordering from English to Spanish will naturally occur when we do not know the ground truth translations.

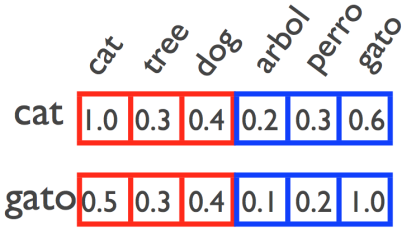

Figure 2: Example of a global vector. The vectors contain the following comparisons from top left to bottom right: (E to E), (E to F), (F to E), and (F to F).

In the final stage of our approach, we identify matches by taking the euclidean distance between our global vectors. We rank the distances in order to find our top-N candidate translations.

2.4 Normalized Edit Distance

The initial scope of our project was combining textual and visual information in order to generate bilingual translations. We have not, however, collected the necessary nonparallel textual information at this point. Thus, as a temporary substituion, we include text information in the form of the normalized edit distance

(NED). This distance is a simple measure of word similarity based on the number of insertions, deletions, and transformations needed to get from one word to the other. For example, the NED between *cat* and *gato* is 0.5, indicating some similarity between the words. The English word *tomato* and Spanish word *tomate* have a very high NED score of 0.83, as only one character varies. Initial experiments on taking a linear combination of SIFT features and NED scores proves very fruitful in aiding performance.

3 Results

In this section, we present the preliminary results of our approach, on English to Spanish translations for a lexicon of 500 words, with k=1000. We have replicated the results of [1] for k=10000, but have not yet produced results on our global vector. We are similarly in the process of scaling up to include results on French, German, Italian, and Dutch translations.

Figure 3 shows results on a 100 and 500-word lexicon, L. We observe an expected increase in accuracy when using a smaller lexicon size. To represent proportional accuracy metrics, we show the top-1, top-5, and top-20 results for L=500 and the top-1, top-2, and top-5 results for L=100.

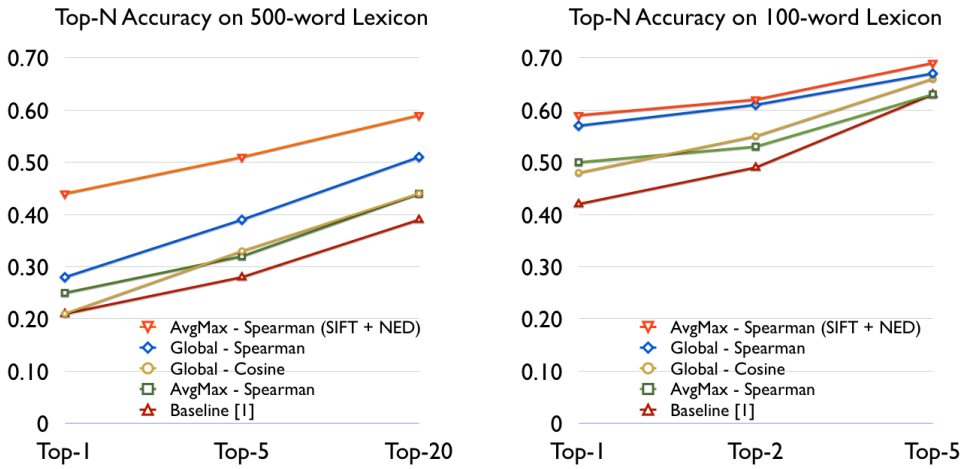

Figure 3: Results on 500-word and 100-word lexicon. Visual word vocabulary is size k=1000.

For both lexicons, all adjustments to the baseline approach from [1] show improvement. The raw baseline results are labeled "Baseline". We refer to the approach used in [1] as "AvgMax", and we indicate the adjustments we applied. Our novel vector approach is labeled as "Global". The results show that measuring the similarity between classes using the Spearman correlation improves accuracy in all cases. We also observe

It is also important to note that the data set from [1] includes near-duplicate images across languages, making the problem less compelling from a vision standpoint. We thus experimented with detecting and removing the near-duplicate images in order to see the accuracy using truly nonparallel data. In figure 4, we show statistics on the near-duplicate images that appear in both the English and Spanish sets. We also show the results of our tests. Removing the near-duplicates has a negative but non-detrimental effect on our results, indicating that we can still achieve strong results with mutually exclusive image sets.

	Spanish		
	Class	Images	% of Class
Top 5	crack	16	94%
	baracudda	15	79%
	fraser	14	82%
	goo	11	58%
	bell	8	42%
Bottom 5	athlete	1	5%
	ladder	1	5%
	corporation	1	5%
	image	1	5%
	actor	1	5%

Total duplicates: 534
Classes with duplicates: 84

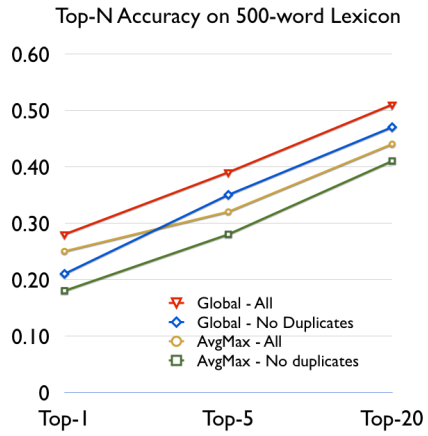

Figure 4: Results from removal of near-duplicates vs. using entire image set.

4 Conclusions and Future Work

By using images in multilingual data sets, we can conduct bilingual translations through extracting and comparing visual features. We observe the increase in accuracy that comes from representing images in a global context. By simply comparing image similarity and identifying the best matches among classes, we lose a significant amount of information, as we fail to consider the similarity of classes to all other classes. This novel approach to constructing feature vectors could have a myriad of applications in the object recognition and classification. We also observe the improved performance by measuring the correlation, as opposed to the distance, of image histograms generated by the bag-of-visual words model.

We expect to implement a variety of extensions on our work. First, we plan to introduce a learning method to this approach by created a model to represent classes. We currently perform image-to-image comparisons to generate translations, but we are working on conducting a model-to-image comparison by creating a model for each class in English and using a K-nearest neighbors classifier to predict results based on the k-nearest images in our foreign set. We also plan to create a model-to-model approach in which we model each class in both languages and subsequently perform a comparison.

References

[1] S. Bergsma, B. Van Durme, Learning Bilingual Lexicons using the Visual Similarity of Labeled Web Images, In Proc. IJCAI 2011.
[2] David G. Lowe. Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision, pages 91-110, January 2004.
[3] Alessandro Bergamo, Lorenzo Torresani. Meta-Class Features for Large-Scale Object Categorization on a Budget. Computer Vision and Pattern Recognition (CVPR), 2012.

Additional Notes

I have begun implementing the model-to-image approach, with kNN classification, as previously discussed, but have not yet gotten results. I have also been working on scaling up to produce results on all images, as well as a visual vocabulary of size k=10,000. While I have working code in python to find the AvgMax comparisons needed to generate our global vectors, this code takes more than 24 hours to run for k=10,000. Thus, I rewrote the code to run in Matlab which takes much less time, but due to the large amount of files and sheer size of the files, I believe that my Matlab process may have a bug when dealing with larger sizes of k and other languages. The pipeline as is contains many thousands of files and scripts, making it very easy to make errors that impede performance. Matlab also has trouble dealing with very large files. All of this considered, I plan to work over the break on creating a better pipeline and process for our approach, by debugging and rewriting code.

In addition, I have conducted preliminary but informal and undocumented tests that show promise for our global vector working on French as well as English. A bug appears to currently exist in dealing with French, and because of the large of amount of manual manipulation, I did not document my results. As mentioned, revision of code over break should help with progression.