Joint Visual-Text Modeling for Multimedia Retrieval

Giridharan Iyengar, Pinar Duygulu, Shaolei Feng, Pavel Ircing,
Sanjeev Khudanpur, Dietrich Klakow, Matthew Krause, R. Manmatha,
Harriet J. Nock, Desislava Petkova, Brock Pytlik, Paola Virga

November 24, 2004

1 Introduction

There has been a renewed spurt of research activity in Multimedia Information Retrieval. This can be partly attributed to the emergence of a NIST-sponsored video analysis track at TREC[1], coinciding with a renewed interest from industry and government in developing techniques for mining multimedia data.

Majority of the state-of-the-art multimedia retrieval systems are a system-level combination of speech-based retrieval techniques and image content-based retrieval techniques. It is our hypothesis that such system-level integration allows only limited exploitation of cues that occur in the different modalities. In addition, techniques used in retrieval systems using images and speech differ vastly and this further inhibits interaction between these systems for multimedia information retrieval. For instance, if the query words have been incorrectly recognized then speech-based retrieval systems may fail. Current systems back-off to image content-based searches and since image retrieval systems perform poorly for finding images related by semantics, the overall performance of such late-fusion systems can be poor. This situation is exacerbated in cross-language information retrieval where there is an additional degradation in the text transcript resulting from machine translation.

In this workshop, we investigate a unified approach to multimedia information retrieval. We represent a multimedia document in terms of visual and textual tokens and build various joint statistical models. This allows us to treat multimedia retrieval as a task of retrieving document comprising visual tokens and words; A generalization of statistical text retrieval models into multimedia retrieval models. With joint visual-text modeling, we demonstrate that we can better represent the relationships between words and the associated visual cues. In this work, we phrase the multimedia retrieval task in terms of a *generative* model. That is, we model the different ways the query q is generated from the document d. We then rank the documents using $p(\mathbf{d}|\mathbf{q})$, or given a query $\mathbf{q}$, the probability that the document $\mathbf{d}$ generated it. To illustrate and validate the usefulness of this approach, we build automatic multimedia retrieval systems, and present experimental results on the TRECVID03 corpus and queries.

2 Retrieval Models

Given a query, $\mathbf{q}$, we want to rank documents, $\mathbf{d}$, according to $p(\mathbf{d}|\mathbf{q})$ as in any other probabilistic information retrieval problem formulation. In our case this can be expanded as below.

$$\begin{aligned} p(\mathbf{d}|\mathbf{q}) &= p(\mathbf{d_w}, \mathbf{d_v}|\mathbf{q_w}, \mathbf{q_v}) \\ &= \frac{p(\mathbf{q_w}, \mathbf{q_v}|\mathbf{d_w}, \mathbf{d_v}) p(\mathbf{d_w}, \mathbf{d_v})}{p(\mathbf{q_w}, \mathbf{q_v})} \end{aligned} \tag{1}$$

In Eq. 1 the denominator can be ignored for ranking documents given any query. In addition, at present we will assume that all documents are equally likely. Any relaxation of this assumption can be done externally and applied to all the models that we develop here. This simplifies Eq. 1 to

$$p(\mathbf{d}|\mathbf{q}) \propto p(\mathbf{q_w}, \mathbf{q_v}|\mathbf{d_w}, \mathbf{d_v}) \tag{2}$$

There may not be enough data to jointly model the above, necessiating the simplifying assumptions. Eq. 2 will get factored into different forms depending on the modeling assumptions made. The first simplification we will make is to assume that the query word tokens and visual tokens (visterms) are conditionally independent given the document. That is the right-hand side of Eq. 2 can be written down as

$$p(\mathbf{q_w}, \mathbf{q_v}|\mathbf{d_w}, \mathbf{d_v}) = p(\mathbf{q_w}|\mathbf{d_w}, \mathbf{d_v}) \times p(\mathbf{q_v}|\mathbf{d_w}, \mathbf{d_v}) \tag{3}$$

2.1 Linear Mixture Model

Consider the term $p(\mathbf{q_w}|\mathbf{d_w}, \mathbf{d_v})$. We can choose to approximate it with a linear mixture model:

$$p(\mathbf{q_w}|\mathbf{d_w}, \mathbf{d_v}) = \lambda_w p(\mathbf{q_w}|\mathbf{d_w}) + (1 - \lambda_w) p(\mathbf{q_w}|\mathbf{d_v}) \tag{4}$$

Now, each of the two sub-components can be independently estimated using two different models. Another choice is to completely ignore the second term (equivalent to setting the mixture weight $\lambda_w = 1$). We can model the visual term $p(\mathbf{q_v}|\mathbf{d_w}\mathbf{d_v})$ similarly:

$$p(\mathbf{q_v}|\mathbf{d_w}, \mathbf{d_v}) = \lambda_v p(\mathbf{q_v}|\mathbf{d_w}) + (1 - \lambda_v) p(\mathbf{q_v}|\mathbf{d_v}) \tag{5}$$

Putting it all together, we get

$$\begin{aligned} p(\mathbf{q_w}, \mathbf{q_v}|\mathbf{d_w}, \mathbf{d_v}) = {} & (\lambda_w p(\mathbf{q_w}|\mathbf{d_w}) + (1 - \lambda_w) p(\mathbf{q_w}|\mathbf{d_v})) \times \\ & (\lambda_v p(\mathbf{q_v}|\mathbf{d_w}) + (1 - \lambda_v) p(\mathbf{q_v}|\mathbf{d_v})) \end{aligned} \tag{6}$$

In addition, people have typically found that putting different weights on the different modalities usually helps. So, we extend this equation to

$$\begin{aligned} p(\mathbf{q_w}, \mathbf{q_v}|\mathbf{d_w}, \mathbf{d_v}) = {} & (\lambda_w p(\mathbf{q_w}|\mathbf{d_w}) + (1 - \lambda_w) p(\mathbf{q_w}|\mathbf{d_v}))^{\beta(\mathbf{q})} \times \\ & (\lambda_v p(\mathbf{q_v}|\mathbf{d_w}) + (1 - \lambda_v) p(\mathbf{q_v}|\mathbf{d_v}))^{1-\beta(\mathbf{q})} \end{aligned} \tag{7}$$

where $\beta(\mathbf{q})$ is a query-dependent weighting of the different modalities. This will be the most general form that we will be considering. Most models that we will consider can be seen as special cases of this model. We will drop $\beta(\mathbf{q})$ from the equations with the understanding that it is external to this discussion and can always be introduced into every model we detail[1].

2.2 Log Linear Model

Below is a maximum-entropy inspired approach which could be an alternative to the linear model. To simplify matters we will start with the problem of estimating

$$p(\mathbf{d_w}, \mathbf{d_v}, \mathbf{q_w}, \mathbf{q_v}) \tag{8}$$

The full probability is difficult to estimate because of a lack of training data. Hence, we will assume that only pair distributions (e.g. $p(\mathbf{d_w}, \mathbf{d_v})$ or $p(\mathbf{d_w}, \mathbf{q_v})$) can be reliably estimated. This amounts to a set of constraint equations:

[1] IBM's TRECVID 2003 experiments suggest that a good $\beta(q)$ is quite valuable.

$$\sum_{d_w, d_v} p(\mathbf{d_w}, \mathbf{d_v}, \mathbf{q_w}, \mathbf{q_v}) = p(\mathbf{q_w}, \mathbf{q_v}) \tag{9}$$

$$\sum_{d_w, q_w} p(\mathbf{d_w}, \mathbf{d_v}, \mathbf{q_w}, \mathbf{q_v}) = p(\mathbf{d_v}, \mathbf{q_v}) \tag{10}$$

$$\sum_{d_w, q_v} p(\mathbf{d_w}, \mathbf{d_v}, \mathbf{q_w}, \mathbf{q_v}) = p(\mathbf{d_v}, \mathbf{q_w}) \tag{11}$$

$$\sum_{d_v, q_w} p(\mathbf{d_w}, \mathbf{d_v}, \mathbf{q_w}, \mathbf{q_v}) = p(\mathbf{d_w}, \mathbf{q_v}) \tag{12}$$

$$\sum_{d_v, q_v} p(\mathbf{d_w}, \mathbf{d_v}, \mathbf{q_w}, \mathbf{q_v}) = p(\mathbf{d_w}, \mathbf{q_w}) \tag{13}$$

$$\sum_{q_w, q_v} p(\mathbf{d_w}, \mathbf{d_v}, \mathbf{q_w}, \mathbf{q_v}) = p(\mathbf{d_w}, \mathbf{d_v}) \tag{14}$$

Using a maximum entropy approach a probability distribution can be found that satisfies all six constraints. Instead of doing a full maximum entropy approach, we will just do one iteration of generalized iterative scaling (GIS).
Assuming statistical independence of all four random variables the initial distribution is:

$$p_0(\mathbf{d_w}, \mathbf{d_v} \mathbf{q_w}, \mathbf{q_v}) = p(\mathbf{d_w}) p(\mathbf{d_v}) p(\mathbf{q_w}) p(\mathbf{q_v}) \tag{15}$$

After one iteration of GIS we arrive at:

$$\begin{aligned} p_1(\mathbf{d_w}, \mathbf{d_v}, \mathbf{q_w}, \mathbf{q_v}) = \frac{1}{Z} p_0(\mathbf{d_w}, \mathbf{d_v}, \mathbf{q_w}, \mathbf{q_v}) \left(\frac{p(\mathbf{q_w}, \mathbf{q_v})}{p(\mathbf{q_w}) p(\mathbf{q_v})} \right)^{\lambda_1} \\ \left(\frac{p(\mathbf{d_v}, \mathbf{q_v})}{p(\mathbf{d_v}) p(\mathbf{q_v})} \right)^{\lambda_2} \left(\frac{p(\mathbf{d_v}, \mathbf{q_w})}{p(\mathbf{d_v}) p(\mathbf{q_w})} \right)^{\lambda_3} \left(\frac{p(\mathbf{d_w}, \mathbf{q_v})}{p(\mathbf{d_w}) p(\mathbf{q_v})} \right)^{\lambda_4} \\ \left(\frac{p(\mathbf{d_w}, \mathbf{q_w})}{p(\mathbf{d_w}) p(\mathbf{q_w})} \right)^{\lambda_5} \left(\frac{p(\mathbf{d_w}, \mathbf{d_v})}{p(\mathbf{d_w}) p(\mathbf{d_v})} \right)^{\lambda_6} \end{aligned} \tag{16}$$

where Z is a normalization and the λ_i are weights for the six constraint equations. Ignoring all terms that do not matter for the decision and also assuming a uniform distribution for $p(\mathbf{d_w}, \mathbf{d_v})$ gives:

$$\begin{aligned} p_1(\mathbf{d_w}, \mathbf{d_v}, \mathbf{q_w}, \mathbf{q_v}) \propto (p(\mathbf{d_v}, \mathbf{q_v}))^{\lambda_2} (p(\mathbf{d_v}, \mathbf{q_w}))^{\lambda_3} \\ (p(\mathbf{d_w}, \mathbf{q_v}))^{\lambda_4} (p(\mathbf{d_w}, \mathbf{q_w}))^{\lambda_5} \end{aligned} \tag{17}$$

This can be transformed into

$$\begin{aligned} p_1(\mathbf{q_w}, \mathbf{q_v} | \mathbf{d_w}, \mathbf{d_v}) \propto (p(\mathbf{q_v} | \mathbf{d_v}))^{\lambda_2} (p(\mathbf{q_w} | \mathbf{d_v}))^{\lambda_3} \\ (p(\mathbf{q_v} | \mathbf{d_w}))^{\lambda_4} (p(\mathbf{q_w} | \mathbf{d_w}))^{\lambda_5} \end{aligned} \tag{18}$$

This framework has been tested in language modeling. There, it usually outperformed linear interpolation. It may be an alternative to Eq. 6. Note that it has the same number of free parameters, as one of the exponents can be set to one without any influence on the ranking of the documents. We note here that this approach uses the same component conditional probabilities as in Eq. 6. Whenever appropriate, this model can be used instead of the linear mixture model.

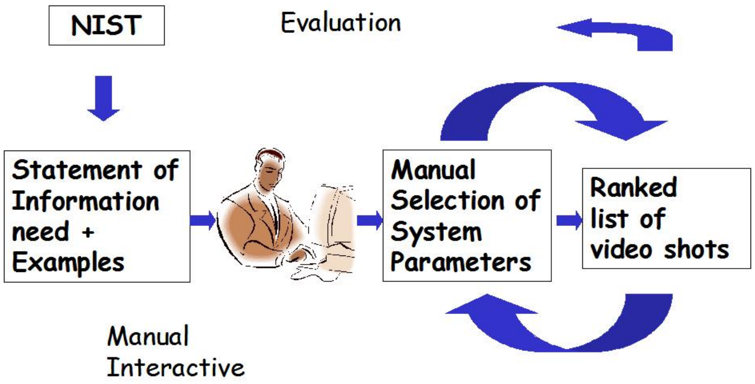

NIST
Evaluation
Statement of Information need + Examples
Manual Selection of System Parameters
Ranked list of video shots
Manual
Interactive

Figure 1: *The manual and interactive system designs permitted by NIST in TRECVID evaluations*

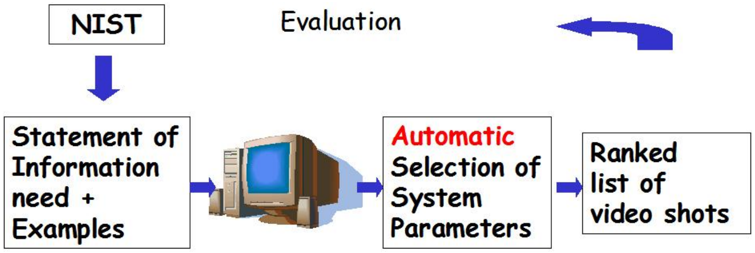

NIST
Evaluation
Statement of Information need + Examples
Automatic Selection of System Parameters
Ranked list of video shots

Figure 2: *Automatic Multimedia Information Retrieval: System design*

3 Baseline System

In multimedia retrieval tasks, text-based systems have outperformed image content-based systems by a wide margin. Therefore, we will compare the joint-modeling based systems with text-based systems. In addition, while NIST permits *manual* and *interactive* query runs where a system operator interprets the query and fires the retrieval system appropriately, we will restrict our experiments to *automatic* systems where there is no human intervention. This choice is to restrict our system design to only the algorithmic issues and ignore the system interface related choices. Figures 1 and 2 illustrate the differences between NIST and our system designs.

In the framework that we propose, the baseline system is obtained by setting $\lambda_w = 1$ and leaving out the visual component. We further assume that all the words in the document are independent of each other given the document, i.e. the bag of words document model. This results in a simple unigram language model over the words in a document[2]. We get

$$p(\mathbf{q_w}, \mathbf{q_v}|\mathbf{d_w}, \mathbf{d_v}) = p(\mathbf{q_w}|\mathbf{d_w}) = \prod_{i=1}^{m} p(q_{w_i}|\mathbf{d_w}) \tag{19}$$

where $\mathbf{q_{wi}}$ are the words in the query. $p(\mathbf{w}|\mathbf{d})$ can be modeled using a variety of smoothing techniques. For illustration, we use the Jelinek-Mercer smoothing to give us

$$p(\mathbf{w}|\mathbf{d}) = \alpha\frac{\#(\mathbf{w},\mathbf{d})}{|\mathbf{d}|} + (\mathbf{1}-\alpha)\mathbf{p}(\mathbf{w}|\mathbf{C}) \tag{20}$$

where $\#(w, d)$ is the number of times the word w occurs in document d and $|d|$ is the total number of words in that document. C is the entire corpus of documents. In addition, we can attempt to relate query words to document words by performing semantic smoothing using a markov chain or estimating a stochastic dictionary using machine translation (see [3, 4] for examples of both approaches). For our baseline, we chose unigram modeling and smoothing with Dirichlet prior as this gave the best results on the test data.

4 Visual Feature extraction

We now detail the visual front-end used during this workshop. The structure of the workshop did not permit extensive experimentation on the visual features used. We detail one set of choices that we made for these experiments and will detail the effect of other feature choices in a later report.

Prior work in the literature(e.g. [5]) suggest a uniform grid partitioning of the image for extracting features from localized regions results in better performance compared to extracting features from image segments. This could be attributed to the current state of image segmentation algorithms. We choose a uniform grid partitioning, specifically a 50×50 pixels partitioning of the image. This gives us a grid with 35 regions on a typical MPEG1 resolution keyframe. In addition, NIST provided us with reference keyframes for the entire TRECVID03 corpus. While we extracted features from both the NIST supplied keyframes and all the I-frames in the MPEG-format videos, we restricted our experiments to the keyframes.

For the *Color* representation, we chose the LAB space [2] moments. For each of the three channels we extract the mean, variance, kurtosis and skewness in that region. This gives a 12-dimensional vector at each region. We detect the edges in the keyframes using the Sobel derivative operator. This gives us an edge strength and orientation at every pixel. These values are quantized into 64 bins (8 strength and 8 orientations) for each of the regions. This histogram is the *Edge* representation of the keyframe regions. The final visual feature that we extracted from the keyframes represents the *Texture* properties of these grid regions. To estimate this feature we transform the keyframe image into 16-levels of grey and compute 4 co-occurrence matrices (horizontal, vertical and the two diagonals). From each of these matrices, we extract summary statistics (namely energy, contrast, inverse difference moment and entropy). For definitions of these statistics see Ref.[6].

5 Relating query words to the visual representation of the document

One possibility to do joint audio-visual retrieval is to build a direct model that relates words to parts of a picture. However, given the present state of computer vision, this is not feasible. Fortunately, TREC-VID data has been annotated with concepts, that cover essential parts of the pictures. Hence, models will be derived, the utilize these concepts.

5.1 Single model with concept layer

In the following, the concepts will be denoted by c. In the previous sections $p(d|q)$ has been decomposed into four different terms for textual and visual queries and the textual and visual parts of the documents.

[2] see the colorspace faq at http://www.faqs.org/faqs/graphics/colorspace-faq/ for a definition of the LAB space.

One of the four terms is $p(q_w|d_v)$ which we will discuss first. To use the concepts, the concept layer is introduced, the probability is decomposed using the definition of conditional probabilities and finally, a kind of Markov or independence assumption is made:

$$\begin{aligned} p(q_w|d_v) &= \sum_c p(q_w c|d_v) && (21) \\ &= \sum_c p(q_w|c\,d_v)p(c|d_v) && (22) \\ &\approx \sum_c p(q_w|c)p(c|d_v) && (23) \end{aligned}$$

Some observations:

- $p(c|d_v)$ is one of the models trained anyway only that the concept labels are used instead of the words.
- $p(q_w|c)$ can be derived from the concept tagger
- Approximating $p(q_w|c\,d_v)$ by $p(q_w|c)$ is very crude. If the query is "Allan Greenspan" the concept will be "face" and such a model alone (even if perfect) will then return only faces.
- The same line of reasoning can be applied to the other three components $p(q_w|d_w)$, $p(q_v|d_w)$ and $p(q_v|d_v)$. In each case, the result is a combination of model types already discussed elsewhere in the paper.

5.2 Suggested use of concept layer models

As indicated above, the concept layer models can not only be used on their own but they can also provide useful smoothing for other noisy and undertrained models. An example:

$$p(q_w|d_v) = \lambda p_{direkt}(q_w|d_v) + (1-\lambda)p_{concept}(q_w|d_v) \qquad (24)$$

with

$$p_{concept}(q_w|d_v) = \sum_c p(q_w|c)p(c|d_v) \qquad (25)$$

This is a linear interpolation of a model that is trained on the data without concept labels and a model with a concept layer: $p_{concept}(q_w|d_v)$. The model $p_{concept}(q_w|d_v)$ will provide additional supporting evidence in cases where $p_{direkt}(q_w|d_v)$ is disturbed by noise. Also it can be used to give lower weights of documents that are missing essential information (e.g. if the query is "Give me a picture of Allan Greenspan" but the document doesn't show a face it should be suppressed.)

However, we have to be careful here. In traditional language modeling, a similar reasoning can be done e.g. for the combination of trigram LMs and grammar LMs. Such a combination gives a significant improvement however, it will not completely suppress ungrammatical sentences. For the application investigated in this work it means that we cannot expect perfect precision on concept level e.g. suppression of pictures without faces if we ask for a picture of a person.

Note: Instead of a linear interpolation also a log-linear combination can be used.

5.3 Machine Translation Inspired Approaches for Image Annotation

One approach to estimate the probability of the concepts given the visual features of a keyframe ($p(c|d_v)$) is to learn the correspondences between concepts and images. In this approach, the correspondence problem is attacked as the translation of visual features into concepts, anologous to the statistical machine translation.

5.3.1 Motivation

In the image and video collections, the images are usually annotated with a few keywords which describe the images. However, the correspondences between image regions and words are unknown. For example, for an image showing a tiger on the grass, and having the annotated keywords `tiger` and `grass`, it is known that tiger and grass are in the image, but it is not known which region is tiger and which region is grass (Figure 3). With a single image, it is not possible to solve the correspondence problem. However, if there were other images, where the orange stripey region (the region corresponding to tiger) was not with a green region (which correspond to grass) but with something else (e.g. a gray region corresponding to ground, or a blue region corresponding to water), then it would be possible to learn that `tiger` was corresponding to the orange stripey region but not to the green one.

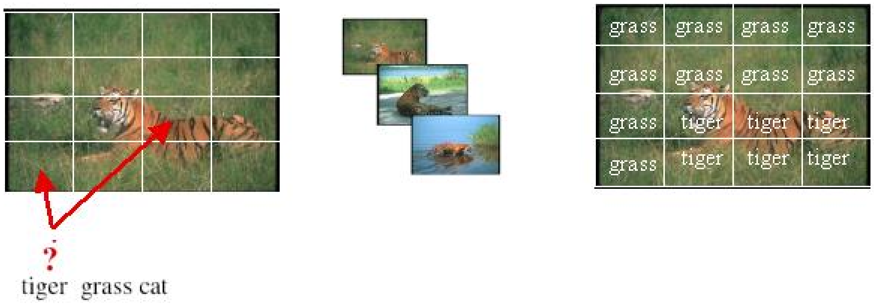


Figure 3: *The correspondence problem between image regions and words: The words* `tiger` *and* `grass` *are associated with the image, but the word-to-region correspondences are unknown. If there are other images, the correct correspondences can be learned and used to automatically label each region in the image with correct words or to auto-annotate a given image.*

This correspondence problem is very similar to the correspondence problem faced in statistical machine translation literature (Figure 4). There are some data sets known as **aligned bitext**, which consist of many small blocks of text in both languages, that are known to correspond to each other at paragraph or sentence level, but word to word correspondences are unknown. A well-known example is the "Hansard Corpus" consisting of debates from the Canadian Parliament, where each speaker's remarks in the country's two official languages -English and French-, correspond in meaning.
Brown *et.al* [7] suggested that it may be possible to construct automatic machine translation systems by learning from such large datasets. Using these aligned bitexts, the problem of lexicon learning is transformed into the problem of finding the correspondences between words of different languages, which can then be tackled by machine learning methods.
Due to the similarity of problems, correspondence problem between image regions and concepts can be attacked as a problem of translating visual features into words, as first proposed by Duygulu *et.al.* [8]. Given a set of training images, it is possible to create a probability table that associates words and visual features which can be then used to find the corresponding words for the given test images.

5.3.2 Approach

In machine translation, a lexicon links a set of discrete objects (words in one language) onto another set of discrete objects (words in the other language). Therefore, in order to exploit the analogy with machine translation, both the images and the annotations need to be broken up into discrete items. The annotation

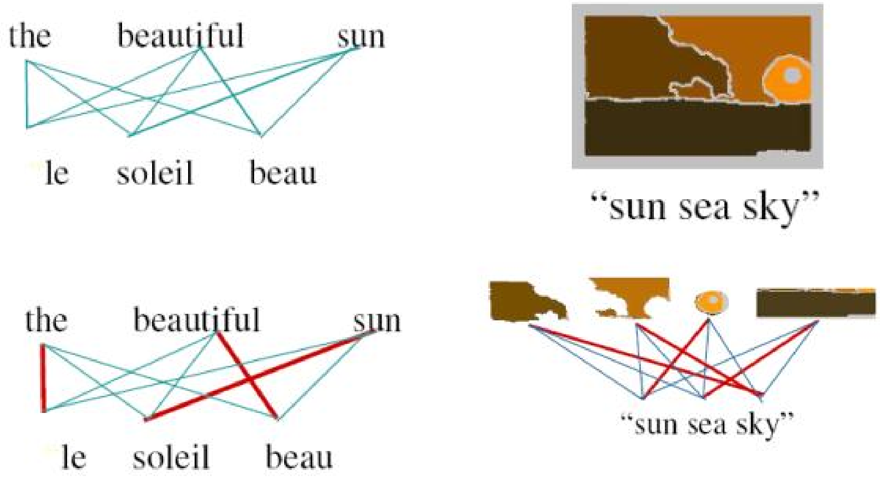


Figure 4: *Correspondence problem between image regions and concepts can be attacked as a problem of translating visual features into words. The problem is very similar to Statistical Machine Translation. We want to transform one form of data (image regions or English words) to another form of data (concepts or French words).*

keywords in Corel data set and the concepts in TRECVID data set can be directly used as discrete items. For data sets, which are annotated in free text form, an appropriate language processing procedure can be applied to reduce the free text annotation into a set of discrete items.

In order to obtain the discrete items for visual data, the images are first segmented into regions. The regions could be obtained by a segmentation algorithm as in [8] or can be fixed sized blocks as we will use in this study. Then, a set of features, such as color, texture, and edge, are computed to represent each region. Finally, the regions are classified into region types (**visterms**) by vector quantization techniques such as K-means.

After having the discrete items, an aligned bitext, consisting of the visterms and the words (concepts in our case) for each image is obtained. The problem is then, to use the aligned bitext in training to construct a probability table linking visterms with concepts.

In this study, we use the direct translation model. Brown *et. al.* [7] propose a set of models for statistical machine translation. The simplest model (Model 1), assumes that all connections for each French position are equally likely. This model is adapted to translate visterms to concepts, since there is no order relation among the visterms or concepts in the data.

The word posterior probabilities for each visterm, supplied by the probability table, is then used to predict concepts for the test data. In order to obtain the word posterior probabilities for the whole image, the word posterior probabilities of the regions in the image, provided by the probability table, are marginalized as given below:

$$P_0(c|d_v) = 1/|d_v| \sum_{v \in d_v} P(c|v) \tag{26}$$

where v's are the visterms in the image. Then, the word posterior probabilities are normalized. Auto-annotation, can be performed by predicting concepts with high posterior probability given the image (Figure 5) .

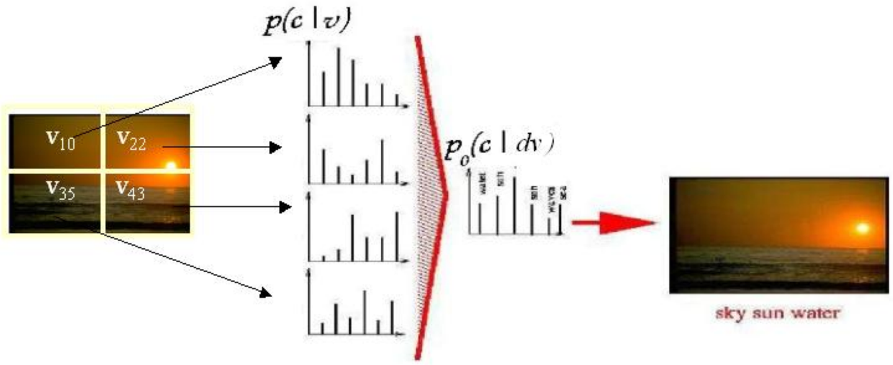


Figure 5: *Auto-annotation strategy. Word posterior probabilities for the regions of the image are summed, and normalized. Then the best* n *words with the highest probability are chosen to annotate the image*

5.3.3 Integrating Word Cooccurrences

We incorporate the language modeling in the form of word cooccurrences, since our data sets consist of individual concepts without any order. In our new model, the probability of a concept given an image depends both to the probability of that concept given other concepts, and the probability of other concepts given the image.

$$P_1(c_i|d_v) = \sum_{j=1}^{|C|} P(c_i|c_j)P_0(c_j|d_v) \tag{27}$$

5.3.4 Experimental Results for Machine Translation Approach

Experiments are carried out both on TRECVID data set and on Corel data set. For the experiments on TRECVID data set, color, texture and edge features are extracted from fixed sized blocks and also around interest points which are obtained by a Harris corner detector based algorithm. The feature vectors are separately quantized into 1000 visterms each. The vocabulary consists of 75 concepts. For the experiments on Corel data set, each image is divided into 24 fixed sized blocks and from each region color and texture features are extracted to form a single feature vector. These feature vectors are then vector quantized into 500 visterms using K-means algorithm. The vocabulary of training set consists of 374 words. Translation tables are learned using Giza++, which is a part of Statistical Machine Translation toolkit developed during summer 1999 at CLSP at Johns Hopkins University.

Figure 6 shows some auto-annotation examples using Model 1 training. Most of the words are predicted correctly and most of the incorrect matches are due to the missing manual annotations (*e.g.* Although `tree` is in the image on the top-left example it is not in the manual annotations.

In order to test the effect of different models we have trained our system also with more complicated models: (i) using HMM on top of Model 1 and, (ii) Model 4 on top of Model 1 and HMM training. The experiments show that, the simplest model (only Model 1) produces the best annotation performance. The Mean Average Precision values obtained by Model 1 are 0.125 on Corel data set and 0.124 on TRECVID data set.

It is observed that, the number of iterations in Giza training also affects the annotation performance. Although, annotation performance descreases with the increased number of iterations, with less iterations less number of words can be predicted. Due to this tradeoff, number of iterations is set to 5 in the experiments.

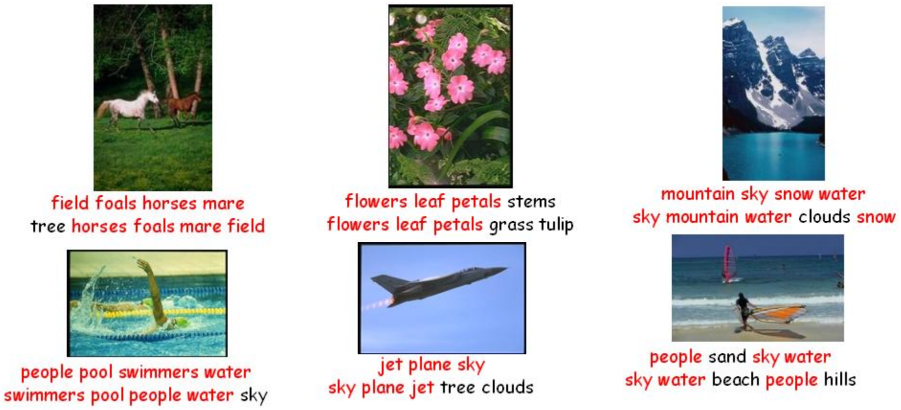

field foals horses mare
tree horses foals mare field
flowers leaf petals stems
flowers leaf petals grass tulip
mountain sky snow water
sky mountain water clouds snow
people pool swimmers water
swimmers pool people water sky
jet plane sky
sky plane jet tree clouds
people sand sky water
sky water beach people hills

Figure 6: *Annotation examples on Corel data set. Top: manual annotations, bottom: predicted words (top 5 words with the highest probability). Words in red color correspond to the correct matches.*

Another important parameter that affects the annotation performance is the features. In the experiments we have compared color, texture, and edge features extracted either from blocks or around interest points. The results are shown in Figure 7. It is observed that, the performance is always better when features are extracted from blocks. The experiments show that, color feature gives the best performence when used individually but using a combination of all three features gives the best performence. Adding a feature related to detected faces (number of faces) does not give any significant improvement. Feature selection based on Information Gain is also experimented, but the results were not satisfactory.
It is shown that (Table 1) incorporating word cooccurrences into the model helps to improve annotation performence for Corel data set, but does not create a difference for TREC data set.

	Corel	TREC
Model 1	0.125	0.124
Model 1 with word cooccurrences	0.145	0.124

Table 1: The effect of incorporating word co-occurrences

Another experiment that has been studied but not performing well was using the alignments provided by training to construct a co-occurrence table. For this experiments we have trained Giza in both ways, i.e. one table is created for co-occurrences by training from visterms to concepts and another one is created by training from concepts to visterms. A third co-occurrence table is created by summing up the two tables. As shown in Table 2, the results were worse than the base results.

Model 1	Alignment(Visterm to Concept)	Alignment(Concept to Visterm)	Alignment (Combined)
0.125	0.103	0.107	0.114

Table 2: Comparison of the results obtained from a co-occurrence table of the alignment counts with the basic Model 1 results.

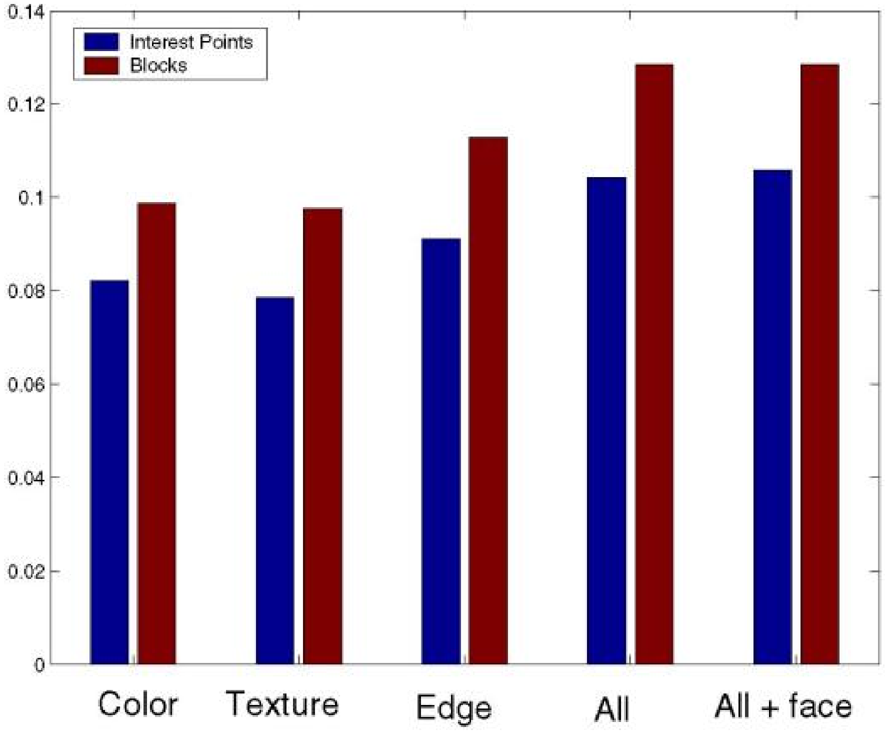


Figure 7: *Comparison between block-based features and Harris interest point features.*

5.3.5 Image Annotation using Cross-Lingual Information Retrieval

The image annotation problem can be viewed as the problem of Cross-Lingual Information Retrieval (CLIR). In CLIR we have queries in a language "A" and the document collection in a language "B". The goal is to find the most relevant documents in language B for each query Q from language A. If we assume that language A is the language of concepts and B is the language of visterms, the task of image annotation becomes a CLIR problem. Suppose we would like to find for the concept c the most relevant images in our collection, we would rank each document using the following equation:

$$p(c|d_V) = \alpha(\sum_{v \in d_V} p(c|v)p(v|d_V)) + (1-\alpha)p(c|G_C), \tag{28}$$

where c is a concept and d_V is a image document. Since the term $p(c|G_C)$ is the unigram probability of the concept c estimated on training data and does not depend on d_V, it will be dropped and the above formula can be rewritten as:

$$p(c|d_V) = \sum_{v \in d_v} p(c|v)p(v|d_V). \tag{29}$$

In order to compute $p(c|d_V)$ we need to estimate $p(v|d_V)$ and $p(c|v)$. The probability $p(v|d_V)$ is computed directly from the document d_V. The probability $p(c|v)$ is the probability of the concept c given that the visterm v is the document d_V; this is obtained as the translation probability estimated in the machine

translation approach. As already mentioned each document is represented by a fixed number of visterms; 105 visterms for the TREC collection is pulled out from visterm vocabulary of size 3000. In this situation $p(v|d_V)$ usually turns out to be close to $\frac{1}{105}$ for each visterm $v \in d_V$.
Since individual images were not able to produce a good estimate of $p(v|d_V)$, we choose to estimate the prior probability over the training collection in the following ways:

$$TF_{Train}(v) = \frac{\#\text{ of v in the collection}}{\#\text{ of visterms in the collection}}$$

$$DF_{Train}(v) = \frac{\#\text{ of documents with v}}{\#\text{ of documents in the collection}}$$

Since document frequency (DF) outperforms the term frequency (TF), we used $DF_{Train}(v)$ as a estimate of $p(v)$. Using $p(v)$ and restricting the sum over only the visterms in the given document, we now have a score that is not a probability:

$$score(c|d_V) = \sum_{v \in d_v} DF_{Train}(v)p(c|v) \tag{30}$$

The annotation performance of the CLIR approach in terms of mAP is 0.126 which is significantly better than our baseline Model 1 (p=0.04).
Figure 8 compares the basic machine translation based approach with CLIR based approach using average precision values for the top 10 words. The recall-precision performance for CLIR is given in Figure 9.

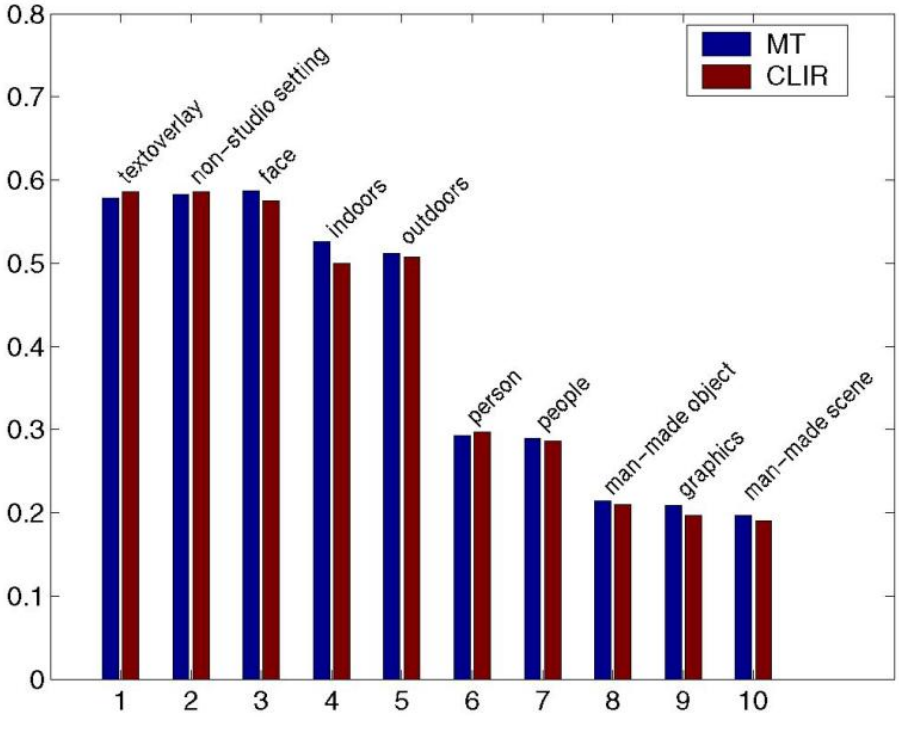


Figure 8: *Average Precision comparison between MT and CLIR based models for the top 10 concepts*

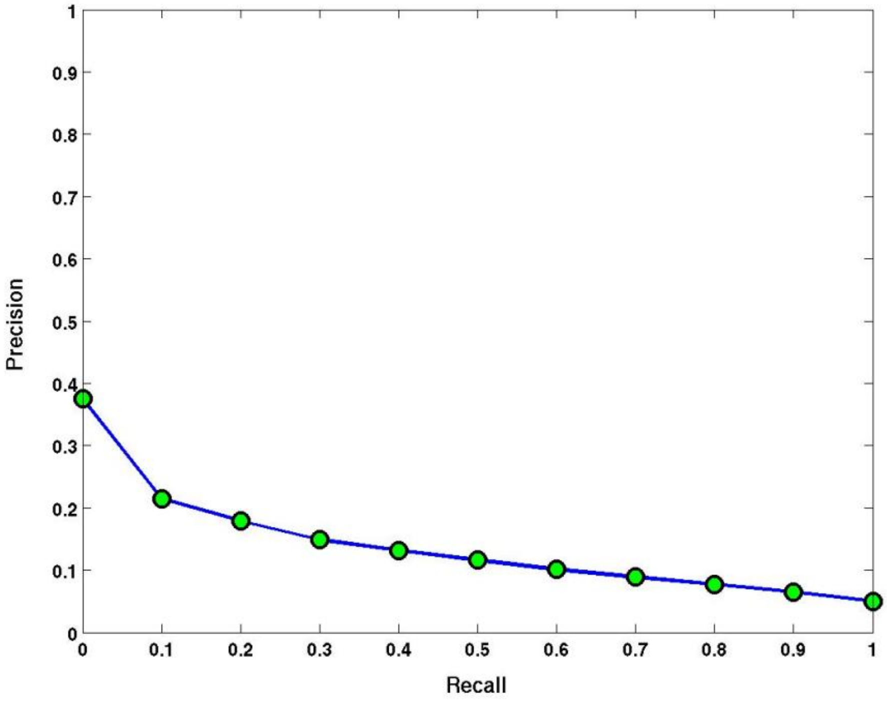


Figure 9: *Recall Precision performance for the CLIR annotation model*

5.4 Relevance Models for Visual Annotation

Relevance models are one approach to estimating the probabilities of annotating a keyframe with concepts given the visual features of a keyframe. The basic idea behind relevance models is to automatically associate words with an image by estimating the joint probabilities of words and the set of visterms obtained from a particular image. The model requires a training set of annotated images.

5.4.1 Motivation

In an image, isolated pixels and regions are hard to interpret. The association of different image regions provides the context of that image, which simplifies the recognition of distinct regions as specific objects. Thus, for example, a region of black stripes over orange is more likely to be associated with the word tiger if it seen with image regions of forest or trees than with regions of a kitchen. So the intuition is that if a model learns the context of images well it works well on image annotation. Given annotated images, one can assume that every image is described either by its annotations (words) or by visterms from an image vocabulary of visterms. A relevance model is a nice way to integrate context information in images through computing the joint probabilities of visterms associated with an image and the associated annotations (concepts in this case).

Relevance models for images were developed in analogy with cross-lingual retrieval [9] where the idea is retrieve documents in one language (say French) given a query in another language (say English). The relevance model approach to this problem is to compute a relevance model which is the joint probabilities of words in two languages for the relevant set from a training set of corpora in two languages. Here, the

essential idea is that every document may be represented using words from two different languages. The relevance model may then be used to retrieve documents in French given English queries.
The first extension to images, the cross-media relevance model [5] assumed that each image may be described using two vocabularies - an image vocabulary of visterms - and a word vocabulary for the annotations. Given a training set of annotated images, this joint distribution of visterms and words may be learned and then used to annotate test images which have only an image description. Discrete visterms are computed as follows: First an image is partitioned into regions using either a segmention algorithm or a regular grid. A feature vector is extracted for each region. K-means clustering of these regions across images creates a set of clusters - the vocabulary of visterms.
Instead of discrete visterms, one may use a continuous version of the model where each region of the image is represented using continuous features. A kernel density estimate is used instead of the discrete representation as described below. Such a continuous relevance model [10] produces better results as described below. Previous work [11] has also shown that a rectangular regular partition produces better results than using a segmentation algorithm. Note that the training data does not provide region/word alignments. However, the relevance model does not create such alignments either (unlike the translation model and HMM's). Region/word alignments are not required for retrieval.

5.4.2 Cross-media Relevance Models

The cross-media relevance model (CMRM) was introduced by Jeon, Lavrenko and Manmatha [5] for annotating images. Our formulation here is based on their original formulation.
A collection of annotated images as a training set, provides the CMRM with two parallel vocabularies: the visterm set from clustering visual features extracted from each image region and the concept words from the human annotation. Unlike machine translation models which are based on one visterm to one word translation for a test image, CMRM uses a probability distribution to specify how often we expect to see any concept words relevant to the visterms from the test image. Thus CMRM implicitly integrates the techniques of translation disambiguation and query expansion.
Given a concept c and the set of visterms $d_v = v_1, v_2, ..., v_n$ from a test image, then we would like to compute the probability of

$$P(c|d_v) = P(c|v_1, v_2, ..., v_n) \tag{31}$$

The conditional probability can be computed if we know the joint distribution $P(c, d_v)$ because

$$P(c|v_1, v_2, ..., v_n) = \frac{P(c, v_1, v_2, ..., v_n)}{\sum_c P(c, v_1, v_2, ..., v_n)} \tag{32}$$

Let τ be the training set then the joint probability of any concept c and the set of visterms $d_v = v_1, v_2, ..., v_n$ from a test image could be computed as an expectation over the images $J \in \tau$:

$$P(c, d_v) = \sum_{J \in \tau} P(J) P(c, v_1, v_2, ..., v_3|J) \tag{33}$$

Given a training image J, we may assume that c and $v_1, v_2, ..., v_3$ are mutually independent. So we can rewrite the equation (33) as:

$$P(c, d_v) = \sum_{J \in \tau} P(J) P(c|J) \prod_{v_i \in d_v} P(v_i|J) \tag{34}$$

The prior probabilities $P(J)$ for all the training images J are kept uniform over the training set τ. Maximum-likelihood estimates are used for the probabilities of concept and each visterm generated from the training image, and the estimates are smoothed with the collection frequencies.

$$P(c|J) = (1-\alpha_J)\frac{\sharp(c,J)}{|J|} + \alpha_J\frac{\sharp(c,\tau)}{|\tau|} \tag{35}$$

$$P(v_i|J) = (1-\beta_J)\frac{\sharp(v_i,J)}{|J|} + \beta_J\frac{\sharp(v_i,\tau)}{|\tau|} \tag{36}$$

where α_J and β_J are smoothing parameters for concepts and visterms respectively, and are selected empirically on a held-out portion of the training set τ. Annotation involves computing the probabilities for each concept.

5.4.3 Continuous Relevance Models

The continuous relevance model(CRM) is a continuous version of the cross-media relevance model. This model also relies on a training set of annotated images and operates as follows. First, we partition each training image into regions using an unsupervised segmentation algorithm[12] or using rectangular partitions. Then, we compute a real-valued feature vector for each region. The features reflect the relative position of a region in the image, its shape, color and texture. As a result, each training image is represented as a set of feature vectors $\mathcal{V} = \{v_1 \ldots v_n\}$ along with a set of concept words $\mathcal{C} = \{c_1 \ldots c_m\}$. As a final step, we construct a joint probability distribution $P(\mathcal{V}, \mathcal{C})$ over the concept words $\mathcal{C}$ and image features $\mathcal{V}$. This joint distribution allows us to find the most likely annotations for new unlabelled images by searching for concept words $\mathcal{C}$ that maximize the conditional probability $P(\mathcal{C}|\mathcal{V}) = P(\mathcal{C}, \mathcal{V})/P(\mathcal{V})$.
Given a test image J represented as d_v, CRM computes the probability $P(c, d_v)$, just like CMRM (equation (33) and equation (34)), as a joint expectation over the space of distributions $P(\cdot|J)$ defined by annotated images J the training set τ. So both CRM and CMRM rely directly on individual images in the training set, allowing the data to speak for itself and avoiding making a-priori assumptions about the structure of the space. Both of them are doubly non-parametric statistical models.
Nevertheless, CRM directly models and takes advantages of continuous features. First, it doesn't suffer from any of the usual difficulties with clustering, such as the cluster granularity and clustering errors. Second, continuous features give a more expressive representation of an image. Using a Gaussian kernel instead of clustering to formulate the similarity of features avoids the hard decision on features' categories. These significant differences between CRM and CMRM lead to substantial improvements in performance of CRM over CMRM.
Now let $\mathbf{v}_A = \{v_1 \ldots v_{n_A}\}$ denote the feature vectors of some image A, which is not in the training set τ. Similarly, let $\mathbf{c}_B$ be some arbitrary subset of concept vocabulary $\mathcal{C}$. We would like to model $P(\mathbf{v}_A, \mathbf{c}_B)$, the joint probability of observing an image defined by $\mathbf{v}_A$ together with annotation concepts $\mathbf{c}_B$. We hypothesize that the observation $\{\mathbf{v}_A, \mathbf{c}_B\}$ came from the same process that generated one of the images J^* in the training set τ. However, we don't know which process that was, and so we compute an expectation over all images $J \in \tau$. The overall process for jointly generating $\mathbf{c}_B$ and $\mathbf{v}_A$ is as follows:

1. Pick a training image $J \in \tau$ with probability $P(J)$
2. Sample $\mathbf{c}_B$ from a multinomial model $P(\cdot|J)$.
3. For $a = 1 \ldots n_A$:
    (a) Sample a generator vector v_a from the probability density $P(\cdot|J)$.

Figure 10 shows a graphical dependency diagram for the generative process outlined above. We show the process of generating a simple image consisting of three regions and a corresponding 3-word annotation.

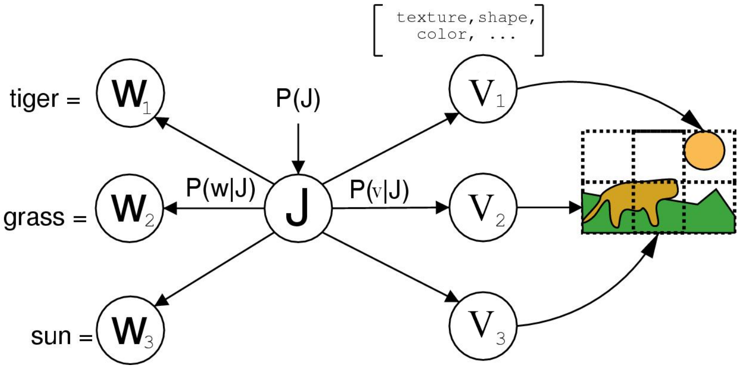


Figure 10: *CRM as a process for generating annotated images. First, pick a training image J. Then, sample the annotation words $c_1 \ldots c_n$ from the multinomial distribution $P(c|J)$. Finally, sample image regions $v_1 \ldots v_n$ from the density function $P(v|J)$*

Note that the number of words in the annotation n_B does not have to be the same as the number of image regions n_A.

As CMRM, CRM computes the concept probability $P(c|J)$ of equation (34) using a maximum likelihood approach smoothed over the training set τ:

$$P(c|J) = \lambda \frac{N_{c,J}}{N_J} + (1 - \lambda) \frac{N_c}{N} \tag{37}$$

Here $N_{c,J}$ is the number of times c occurred in the annotation of J, N_J is the length of annotation, N_c is the total number of times c occurred in the training set, and N is the aggregate length of all training annotations. λ denotes a parameter that controls the degree of smoothing.

CRM models the visterm (which is now a real-valued feature vector) component $P(v|J)$ using a density function responsible for modelling the d-dimensional feature vectors $\{v_1 \ldots v_{n_A}\}$, which are computed from the rectangular regions of each image. We use a non-parametric kernel-based density estimate for the distribution $P(v|J)$. Let $\mathbf{v}_J = \{v_1 \ldots v_n\}$ be the set of regions of image J. We estimate the probability density for a new vector v as:

$$P(v|J) = \frac{1}{n} \sum_{i=1}^{n} \frac{\exp\left\{(v - v_{Ji}))^{\top} \Sigma^{-1} (v - v_{Ji}))\right\}}{\sqrt{2^d \pi^d |\Sigma|}} \tag{38}$$

Equation (38) arises out of placing a Gaussian kernel over the feature vector $v_J i$ of every region of image J. Each kernel is parameterized by the feature covariance matrix Σ. As a matter of convenience we assumed $\Sigma = \beta \cdot I$, where I is the identity matrix. β plays the role of kernel *bandwidth*: it determines the smoothness of $P(v|J)$ around the support points v_i. The value of β is selected empirically on a held-out portion of the training set τ.

5.4.4 Normalized Continuous Relevance Models

One assumption of CRM is the multinomial word distribution, which make the model ill-suited for image/video annotation. CRM assumes that annotation words for any given image follow a multinomial distribution. This is not too unreasonable an assumption in the Corel dataset, where all annotations are approximately equal in length and words reflect the *prominence* of objects in the image. However, in our video datasets individual frames have hierarchical annotations which do not follow the multinomial distribution. The length of the annotations also varies widely for different video frames. Furthermore, video annotations focus on the *presence* of an object in a frame, rather than its *prominence.*

The multinomial model is meant to reflect the prominence of words in a given annotation. The event space of the model is the set of all *strings* over a given vocabulary, and consequently words can appear multiple times in the annotation. In addition, the probability mass is shared by all words in the vocabulary, and during the estimation process the words compete for this probability mass. As a result, an image I_1 annotated with a single word "face" will assign all probability mass to that word, so $P(\text{face}|I_1) = 1$. If some other image I_2 is annotated with ten different words, one of which is "face", we get $P(\text{face}|I_2) = \frac{1}{10}$. Arguably, both images contain "face" in their annotations, so the probabilities should not differ by an order of magnitude.

We can modify the assumption by using a normalized continuous relevance model(Normalized-CRM) [13], which bears the same mathematical framework with CRM except for using a normalized multinomial for word distributions. The normalized CRM first expands all annotations to a fixed length $N^* = \max_J\{N_J\}$, where N_J is the annotation length of image J. This is accomplished by adding $(N^* - N_J)$ instances of a special *"null"* word to the annotation of image J. The word probabilities are estimated using equation (37). The normalized-CRM may be shown to be equivalent to multiple-Bernoulli Relevance Model(MBRM) for annotation performance [11, 14], which uses a multiple-Bernoulli distribution for the concept word probabilities.

5.4.5 Experimental Results for Relevance Models

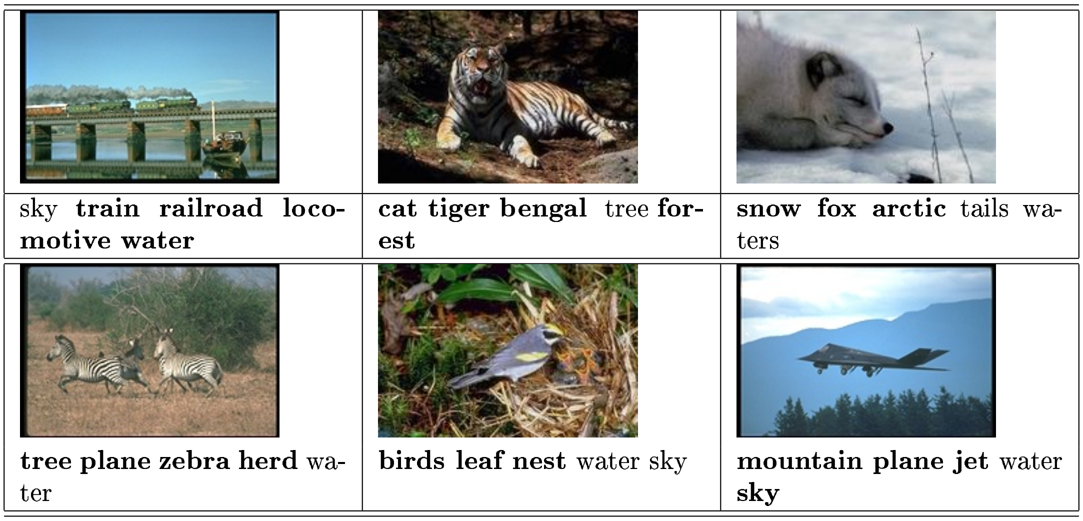
On the alternative dataset-Corel set, we tested all these three kinds of Relevance models: CMRM, CRM and normalized-CRM, and compare their annotation performance. We partition each image in this collection into 24 rectangles and extract visual features (color and texture) from each rectangular region as a visterm. For CMRM, we first classify all the visual features into 500 categories using K-means as visterms. Since K-means uses randomly selected sample points as the initial category centers and cannot guarantee a global optimum, we try the CMRM on different sets of visterms from separate K-means runs to see the effects of the clustering on the model. The results show that even with the same number of categories the clustering does affect the CMRM's performance.

Table 3 show the annotation performance of CMRM, CRM and Normalized-CRM on the Corel dataset. Normalized-CRM works best.

Models	CMRM	CRM	Normalized-CRM
Mean Average Precision	0.14	0.23	0.26

Table 3: Performance comparison of annotation on the Corel dataset

Figure 11 shows some automatic annotation examples for the normalized-CRM on the Corel dataset. Note that normalized-CRM correctly predicates most annotation words for these images, even those missed by human annotation, e.g. "water" for the first image, 'tree' for the second image but also makes some mistakes. For the TRECVID2003, we tested normalized-CRM on the development dataset. The feature set we used only includes color and texture for the sake of computation expenditure since our comparison experiments on a relative smaller set show adding edge features only gives very slight improvement – only 0.01 for the mean average precision. The model's parameters, including the bandwidth for the Gaussian kernel and the smoothing parameter for concept probabilities, are selected by holding out a portion of the training set for validation. The mean average precision obtained by normalized-CRM over the development dataset for annotation is 0.158.



sky **train railroad locomotive water**

cat tiger bengal tree **forest**

snow fox arctic tails waters

tree plane zebra herd water

birds leaf nest water sky

mountain plane jet water **sky**

Figure 11: Top automatic annotations produced by the normalized-CRM models. The bold words are those that appear in the human annotation for that image

Figure 12 shows the comparison of the recall-precision graphs from normalized-CRM and model 1 of the IBM translation model. The normalized-CRM is substantially better than the translation model for low recall.

5.4.6 Related Relevance Models

Intuition All our relevance models in the previous sections make an assumption that the visterms are independent of the concepts given the image, i.e. $P(v|J,c) = P(v|J)$ noting the equation 34. This is a direct analogy with unigrams in text retrieval (bigrams have not shown any significant improvement over unigrams in in text retrieval). One can try more complex models which make the visterms depend on the concepts. However, as the models below show the performance is worse. For example, one can rewrite equation 34 as:

$$P(c, d_v) = \sum_{J \in \mathcal{T}} P(J)P(c|J) \prod_{v_i \in d_v} P(v_i|J, c) \tag{39}$$

To formulate the $P(v|J,c)$, we tried different ideas.

Idea One Approximate the $P(v|J,c)$ with $P(v|c)$, which is computed from the translation models. Now the model become:

$$P(c, d_v) = \sum_{J \in \mathcal{T}} P(J)P(c|J) \prod_{v_i \in d_v} P(v_i|c) \tag{40}$$

In this case the relevance model reduces to a translation model plus a language model, because in equation (40), $\prod_{v_i \in d_v} P(v_i|c)$ is unrelated to the image J and thus plays the role of a translation model, and the rest of the equation plays the role of a language model. To make this more explicit, we may rewrite it as:

$$P(c, d_v) = \prod_{v_i \in d_v} P(v_i|c) \sum_{J \in \mathcal{T}} P(J)P(c|J) \tag{41}$$

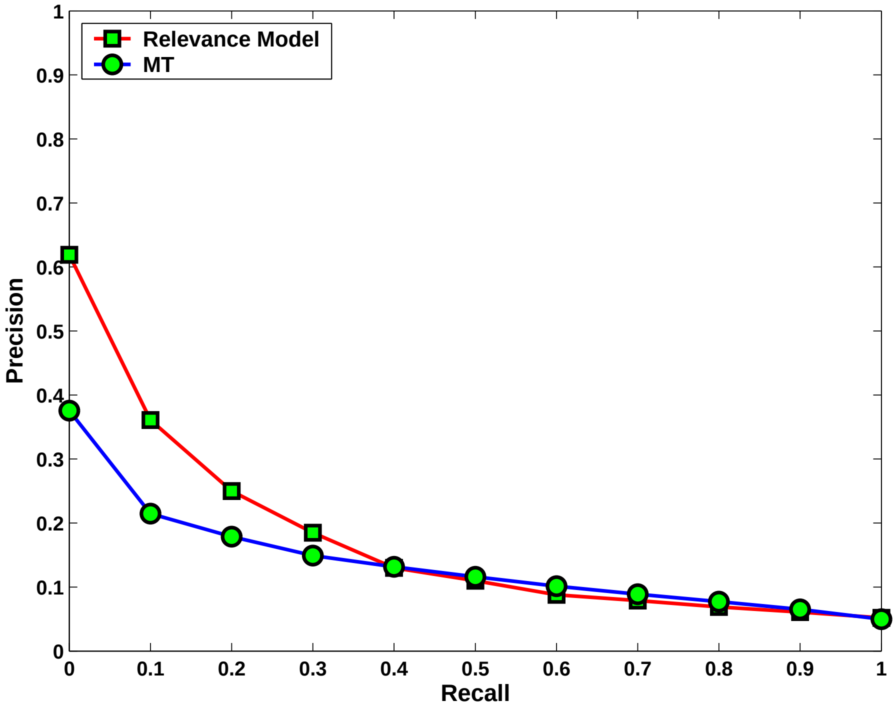


Figure 12: *The comparison of the recall-precision graphs from normalized-CRM and translation models*

Idea Two Use a linear combination of $P(v|J)$ and $P(v|c)$ to approximate $P(v|J,c)$, i.e. $P(v|J,c) = \lambda P(v|J) + (1-\lambda)P(v|c)$, where λ is a parameter determining the weights of the combination, $P(v|J)$ is calculated with a Gaussian kernel as in equation (38) and $P(v|c)$ using the results from the translation model. So the model becomes:

$$P(c, d_v) = \sum_{J \in \mathcal{T}} P(J)P(c|J) \prod_{v_i \in d_v} (\lambda P(v_i|J) + (1-\lambda)P(v_i|c)) \tag{42}$$

Note that, v_i are in different status in $P(v_i|J)$ and $P(v_i|c)$: v_i are real-valued visterm features in $P(v_i|J)$ while they are discrete visterms in $P(v_i|c)$.

Idea Three In equation (38), each Gaussian kernel function essentially formulates the probability of the test visterm v given each visterm v_{Ji} of the image J. So we can write equation 38 as:

$$P(v|J) = \frac{1}{n} \sum_{i=1}^{n} P(v|v_{Ji}) \tag{43}$$

where n is the number of visterms in the image J.
After adding the independence on concepts, we have:

$$P(v|J,c) = \frac{1}{n} \sum_{i=1}^{n} P(v|v_{Ji}, c) \tag{44}$$

To compute $P(v|v_{Ji}, c)$, we use Bayes' rule:

$$\begin{aligned} P(v|v_{Ji}, c) &= \frac{P(v, c|v_{Ji})}{P(c|v_{Ji})} \\ &= \frac{P(c|v, v_{Ji})}{P(c|v_{Ji})} P(v|v_{Ji}) \end{aligned} \tag{45}$$

We use co-occurrence tables to compute $P(c|v, v_{Ji})$ and $P(c|v_{Ji})$:

$$P(c|v, v_{Ji}) = \frac{\sharp(c, v, v_{Ji})}{\sharp(v, v_{Ji})} \tag{46}$$

$$P(c|v_{Ji}) = \frac{\sharp(c, v_{Ji})}{\sharp(v_{Ji})} \tag{47}$$

where $\sharp(c, v, v_{Ji})$, $\sharp(v, v_{Ji})$ and v_{Ji} are the counts of (c, v, v_{Ji}),(v, v_{Ji}),v_{Ji} in the training set respectively, and v,v_{Ji} are discrete visterms.
For $P(v|v_{Ji})$ we still use the Gaussian kernel, so here v and v_{Ji} are real-valued visterms.
Now the model becomes:

$$P(c, d_v) = \sum_{J \in \mathcal{T}} P(J)P(c|J) \prod_{v_i \in d_v} (\frac{1}{n} \sum_{i=1}^{n} (\frac{P(c|v, v_{Ji})}{P(c|v_{Ji})} P(v|v_{Ji}))) \tag{48}$$

5.4.7 Experimental Results and Analysis

We implemented these three models and tested them on the Corel dataset. Unfortunately, none of these models outperform the previous relevance models. Although the reasons are not completely clear, it is possible that either we do not have enough data to estimate these dependencies accurately or like in text retrieval, unigrams perform better on this task (given that the HMM's performed better when the transition probabilities were uniform, it is likely that the latter reason may hold).

Unconnected Model	
Standard Model	$p(c_t) = \frac{1}{\lvert C \rvert}$
One-Dimensional Model	
Standard Chain Model	$p(c_t) = p(c_t \mid c_{t-1})$
Two-Dimensional Model	
Full Dependence Model	$p(c_t) = \hat{p}(c_t \mid c_{t-1}, c_{t-S_R}) \vee t \text{ s.t } t \mod S_R > 0 \wedge t - S_R \geq 0$ $p(c_t) = \hat{p}(c_t \mid c_{t-1}) \vee t \text{ s.t } t \mod S_R > 0 \wedge t - S_R < 0$ $p(c_t) = \hat{p}(c_t \mid c_{t-S_R}) \vee t \text{ s.t } t \mod S_R = 0 \wedge t - S_R \geq 0$ $p(c_t) = \hat{p}(c_t) \vee t \text{ s.t } t \mod S_R = 0 \wedge t - S_R < 0$

Table 4: Probability Formulae

5.5 Dynamic Bayesian Networks and Hidden Markov Models for Image Annotation

Graphical models were another approach we used to find $p(c|d_v)$. Graphical models offer a greater freedom to express dependencies between concepts. First order HMM's allow the current concept to only depend on the previous concept. Graphical models allow the two dimensional spatial information in images to be captured. We chose to use Dynamic Baysian Networks (DBNs) as our graphical models. GMTK[3] was used to build the DBNs.

5.5.1 Concept Transition Models

5.5.2 Unconnected Model

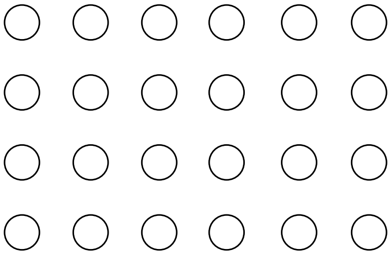


Figure 13: *Unconnected Model*

The unconnected model is the simplest model we have used. Figure 13 depicts the essence of this model.[4] Each visterm is generated by a single concept. Each concept is generated independently with fixed probabilities.[5]

There are several obvious deficiencies in this model. It ignores the spacial organization of the image entirely. It has no way to discover that `sky` is often a large region of the image while `tiger` is often confined to

[3] http://ssli.ee.washington.edu/~bilmes/gmtk/

[4] See Appendix 26 for the full depiction of the model

[5] These probabilities are fixed to be uniform during training. During decoding, they are fixed either to be uniform or according to a language model specified externally.

a smaller region of the image. The subsequent models attempt to address these problems by introducing dependencies between the generation of concepts.

5.6 One-Dimensional Models

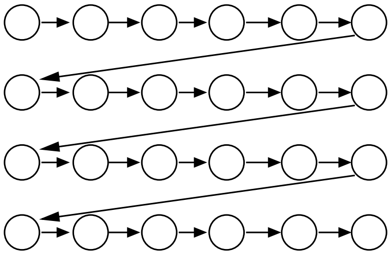


Figure 14: *Chain Model*

The chain model has the generation of each concept depend on the concept to the left of it. Figure 14 has an overview of this model. This model imagines that the image blocks are ordered according to a left to right raster scan of the image and that the concepts generating the blocks are an order 1 Markov chain. This model should capture a notion of width. Concepts like `sky`, which often run from one edge of an image to the other, should have a high probability of generating another `sky` given the current block is `sky`. Concepts like `tiger` or `lichen` should have a lower probability of generating themselves given they are the concept for the current block. It may also be able to capture things like `tiger` is usually to the right of `grass` but not usually to the right of `sky` even though both `tiger` and `sky` often appear in the same image.
This model has attempted to correct for one deficiency of the unconnected model: modeling the size of concepts. It introduces a false dependency in this process. The concept of the left-most block of a row of an image depends on the concept of the right-most of the previous row. Such a dependency is likely undesirable. The model in section 5.6.1 fixes this problem. The simple chain model also fails to take into account any vertical organization an image may have. The models in section 5.6.1 have different approaches to modeling this dependency.

5.6.1 Two-Dimensional Models

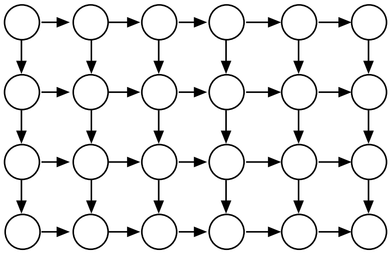


Figure 15: *Full Dependence Model*

Two-dimensional models attempt to capture the vertical as well as horizontal organization of images. Instead of depending solely on the concept to the left, both the concept to the left and above are used to decide the concept of a block. These models form a structure like a grid (see figure 15). These models should be able to capture notions like *if there is* `sky` *above and* `sky` *to the left, this concept should be* `sky` or *if* `grass` *is above and* `tiger` *is to the left, then this concept is probably* `grass` *or* `tiger` *and certainly not* `sky`.

5.6.2 Full Dependence Approach

The straight-forward method of modeling these dependencies is to fully model the horizontal and vertical dependencies of a concept.[6] This model suffers from data sparsity problems in estimating the concept transition probabilities, the space of transitions has gone from $|C|^2$ to $|C|^3$. Gradual training offers one solution to this problem. Initially, a small number of different distributions must be modeled. As more distributions are added, they are initialized with the distribution of the group to which they previously belonged. If few examples for a group of transitions are seen, their distribution could remain unchanged. This would allow a full modeling of common situations, like `sky` above and to the left while not forcing the re-estimation of `lichen` above and `canoe` left.

5.6.3 Training

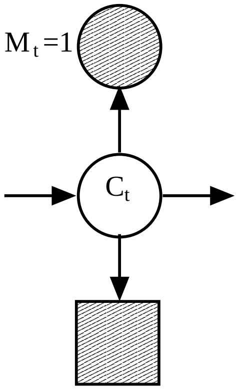


Figure 16: *Only Allow Annotation Concepts*

Only Allow Annotation Concepts One approach to training is to only allow the concepts annotated by a human to be assigned to regions in the image. For example, if an image had labels `sky`, `tiger`, and `grass`, then the model could decide that all image regions were generated by the concept `tiger`. Figure 16 shows how this was implemented. $p(M_t = 1) = 1$ if $C_t \in \{\text{sky, tiger, grass}\}$ and $p(M_t = 1) = 0 \quad \text{otherwise}$.

5.6.4 Force Annotation Concepts to Contribute

The other approach to training is to require each of the annotated concepts to be used at least once. For example, if an image had labels `sky`, `tiger`, and `grass`, then the model could not assign `tiger` to generate all concepts, but could assign `sky` to one region, `tiger` to one region, `grass` to one region, and `ground` to the rest of the regions. Figure 17 shows how this was implemented. In this figure, $M_t^1 = M_{t-1}^1 \vee C_t = \texttt{sky}$, $M_t^2 = M_{t-1}^2 \vee C_t = \texttt{tiger}$, and $M_t^3 = M_{t-1}^3 \vee C_t = \texttt{grass}$. The models trained this way took a great deal longer to train than the other method.

[6] See Table 4 for the specific formula.

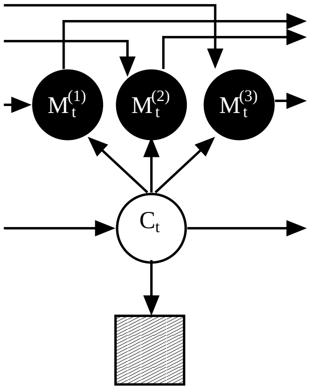


Figure 17: *Force Annotation Concepts to Contribute*

5.6.5 Decoding

Methods for Decoding The first approach to decoding an image we tried was to use the conditional probability of concept given image blocks ($p(c|V)$). We calculated this by first computing the probability of the image blocks ($p(V)$) by calculating the sum of the probabilities of all possible concept sequences for the image blocks. The joint probability of a specific concept and the image blocks is next calculated ($p(c, V)$). We defined $p(c, V)$ to be the sum of the probabilities of all concept sequences for the image blocks where c appears in the sequence at least once. $p(c|V)$ is gotten by dividing $p(c, V)$ by $p(V)$. We refer to this process as forward pass decoding.
A modified approach to decoding uses maxes where sums are used in forward pass decoding. We call this approach viterbi pass decoding. This approach has yielded the best results to this point.

5.6.6 Concept Transition Probabilities

A separate issue in decoding is how the concept transition probabilities are defined. The first approach is to use the probabilities as learnt during training. There may be a danger of these transition probabilities overfitting during training. If the demand annotation scheme (section 5.6.3) is used during training, there is nothing to force concepts to appear, perhaps causing a skew towards very frequent concepts during decoding. A solution to this problem is to use a language model derived from a different method during decoding. One approach is to create a concept cooccurrence matrix from the labeled concepts in the training data. Once normalized, this cooccurrence matrix can be used in place of the concept transition probabilities learned by the model. This method appears to give the best results to this point.

5.6.7 Adjustment to $p(c|d_v)$

It is possible that $p(c|d_v)$ will be 1 for more than 1 c. In that case, it is not clear how to break ties. The method we chose was to order the concepts by $p(c|d_v)$.

5.6.8 DBN Results

Overall, the results thus far for graphical models are disappointing. We were not able to match the performance of the HMM's. Examining only the results within the graphical model framework, the results suggest that greater information is useful. The difference between .086 and .071 has a p-value of .051.

Forward Pass or Viterbi	Cooccurrence Language Model	Adjustment to $p(c\|d_v)$	Training Iterations	mAP
F	N	N	20	.068
F	N	Y	20	.068
F	Y	N	20	.040
F	Y	Y	20	.040
V	N	N	20	**.071**
V	N	Y	20	.069
V	Y	N	20	.067
V	Y	Y	20	.067

Table 5: Unconnected Model Results

Two-dimensional model results were not available. One trend appears to be that the viterbi decoding works better than the forward pass decoding.

5.6.9 HMM experiments

Experiments with the HMM models were carried out using the HTK toolkit [15]. During the training, a separate model is constructed for each training image (frame). The model is fully connected, with states determined by words (concepts) present in the manual annotations. Individual image blocks are then treated as if they were "generated" by those annotation concepts. An alignment between image blocks and annotation concepts represents a hidden variable, the models are trained using the EM algorithm. The output probability distribution $p(d_v|c)$ for a particular concept c is shared across all training images - thus all image blocks from the training set depicting `tiger` contribute to a single probability distribution. A single multivariate Gaussian is used to model the output probability distributions in the baseline system.

The probability of transitions between states was fixed to be uniform during the training. This approach corresponds to the unconnected model described in Section 5.5.2. Even though EM algorithms obviously allows to train also the transition probabilities (which would lead to chain model presented in Section 5.6), we have decided to fix the transition probabilities. The reason was that we wanted to fully explore the power of the visual features themselves, without the effects of the linear dependency (which is anyway slightly questionable, see the picture in Section 5.6). The implementation of two-dimensional models would be also possible, but really complex in the HTK framework.

Once the output distributions are trained, a new fully connected HMM is constructed from all the individual states corresponding to vocabulary concepts and this model is used for the decoding. There are two basic types of the decoding models - first of them has a uniform transition probabilities and the second derives the transition probabilities from the concept co-occurrence language model. We have tested both Viterbi and forward pass decoding (see Section 5.6.5) but, unlike in the GMTK implementation, we have found the forward pass approach to perform better and thus all results reported in this section come from the forward pass decoding.

Having implemented this basic training and decoding setup, we also looked at the quality of the automatic alignment between concepts and image regions during the training. Since there no notion of order in the annotation concepts (the fact that the word `tiger` is listed first does not mean that the tiger appears in the upper left corner), learning of the proper alignment is often hard for the EM algorithm.

We tried to improve the alignment by introducing a **gradual training scenario**. First, we identify a set of concepts that often constitute an image background (sky, grass, water, ...). Then we allow only those "background" states to have their individual emission probability distributions in the initial stages of HMM

Forward Pass or Viterbi	Cooccurrence Language Model	Adjustment to $p(c\|d_v)$	Training Iterations	mAP
F	N	N	20	.031
F	N	Y	20	.053
F	Y	N	20	.048
F	Y	Y	20	.060
V	N	N	20	.066
V	N	Y	20	.073
V	Y	N	20	**.086**
V	Y	Y	20	**.086**
F	N	N	15	.034
F	N	Y	15	.055
F	Y	N	15	.034
F	Y	Y	15	.034
V	N	N	15	.068
V	N	Y	15	.074
V	Y	N	15	.030
V	Y	Y	15	.030

Table 6: Standard Chain Model Using DIscrete Visterms Results

training (all other objects share a single "foreground" distribution). After several EM iterations, we start to gradually untie the "foreground" distribution while running more training iterations.

Although this gradual approach subjectively improved the quality of the alignment, it did not provide a significant gain in terms of the annotation performance.

In another attempt to improve the system performance, we forced the models to visit every state during the training (that is, each annotation concept has to be responsible for at least one image block). This led to huge models and consequently it considerably slowed the training procedure, but the difference in performance was only marginal.

The amount of training data available for individual training concepts of course differs substantially. For some concepts it is possible to train the output probability distributions with many Gaussian mixture components whereas for other concepts we have hardly enough data to train a single Gaussian mixture. Thus we implemented a training procedure that gradually adds mixture components during the training according to the number of occurrences of individual concepts in the manual annotations. This approach yielded a significant improvement of the annotation performance.

Results in terms of mean average precision (mAP) for both Corel and TRECVID data sets are summarized in tables 8 and 9, respectively. All reported results were achieved using one of the simplest training scenarios - that is, neither the gradual untying strategy nor the approach that forces the models to visit every state during the training was used. Note that the Corel database served as a "development" set and therefore we performed more experiments on this data set. Comparison of the Recall-Precision performance of the HMM models with Relevance Models and Machine Translation models is presented in Figure 18. We note here that the input features used for the Machine Translation models and the Relevance models started with the raw features as described earlier in section 4. HMMs trained on this feature did not perform well and the results obtained were close to chance performance. The results presented here are based upon the decorrelated and variance normalized feature set. Noting the improvements to the HMM performance using this feature set, after the workshop we experimented with these features on the MT and RM models. The MT models did not produce a significant change in performance. However, using a small subset of the decorrelated feature

Forward Pass or Viterbi	Cooccurrence Language Model	Adjustment to $p(c\|d_v)$	Training Iterations	mAP
F	N	N	20	.030
F	N	Y	20	.069
F	Y	N	20	.047
F	Y	Y	20	.047
V	N	N	20	.038
V	N	Y	20	**.074**
V	Y	N	20	.045
V	Y	Y	20	.045

Table 7: Standard Chain Model Using Continuous Visterms Results

set, we notice significant improvements to the Relevance Models[7]. The Recall-Precision graph documenting this experiment is shown in Figure 19.

Max. number of mixtures	Language Model	mAP
1	N	0.140
2	N	0.157
4	N	0.157
6	N	0.154
8	N	0.161
10	N	0.161
12	N	0.161
16	N	0.164
20	N	0.161
30	N	0.160
50	N	0.162
1	Y	0.155
50	Y	**0.173**

Table 8: Annotation performance - Corel

6 Relating query visuals to the words in the document

6.1 The Language Model Based Classifier for Concept Annotation

The language model (LM) based classifier trains two language models on the training data. One on the set where the concept is present and the other one on the part of the data where the concept is absent. During testing, both language models are used to calculate perplexity on the test data. The one which gives the smaller preplexity determines the concept assigned to the test data. In principle this is a variant of a Bayes

[7] Relevance Models could not be attempted with the complete decorrelated and variance normalized feature set because of computational limitations.

Max. number of mixtures	Language Model	mAP
1	N	0.094
12	N	**0.145**
100	N	0.142

Table 9: Annotation performance - TRECVID

classifier. Formally that corresponds to

$$\text{argmax}_{c \in c_{present} c_{absent}} \prod_i P(f_i|c)P(c)^\gamma \tag{49}$$

where f_i are the feature from the test set and γ corresponds to the "Language Model Factor" in speech recognition. The probabilities of the language models are smoothed using absolute discounting:

$$P(f_i|c) = \max(\frac{N(f_i, c) - d}{N(c)}, 0) + \frac{dR}{N(c)} \tag{50}$$

with $R = \sum_{i:N(f_i,c)>0} 1$ and d the discounting parameter. Note that $P(c)$ does not need any smoothing.

6.2 Concept Annotation of TRECVID

In this section we will give a couple of statistical properties of concepts that help to build suitable models. Fig. 20 shows the relative frequency of the 75 concepts used in this study. The concepts are sorted by their frequency. Note that only the y-axis is logarithmic and that the data is best fitted by an exponential.

The models introduced in the previous sub-section create some kind of bottleneck. Fig. 20 helps to get an impression of the width of the bottleneck. Naively, one might think, that the number of concepts (here: 75) determines the information that can be passed from the visual to the textual models. However, a closer inspection of the plot shows, that only a few concepts contribute significantly. The self preplexity is 29.7. This is still a relatively high value. Given the fact that each shot has on average 3.8 annotations (from the list of 75!) the set of annotations can still give a relatively accurate account of the content of the image.

An other essential measure on the concepts is the temporal auto-correlation function. This shows how likely it is, that the next shot will have the same annotation given that the present shot has a certain annotation. Fig. 21 gives the auto-correlation function for the two concepts "text-overlay" and "face". Note that the decay is relatively rapid. This is the consequence of the definition of a shot: a relatively strong change in the content of the pictures. This makes concepts short-lived. A consequence of this is that models for concept annotation will only get a weak hint by knowing the annotation of the previous shot. Note, that the concept "monologue" is an exception.

7 Concepts from ASR

Before actually using the various classifiers, we played around with the data. Fig. 22 shows the mean average precision for the five most frequent concepts. It is interesting to observe, that for `text_overlay` and `non-studio_setting` the window size of ±5 shots is probably still too small. On the other hand for `face` the important information for classifying that concept can only found in the present shot and its immediate neighbors.

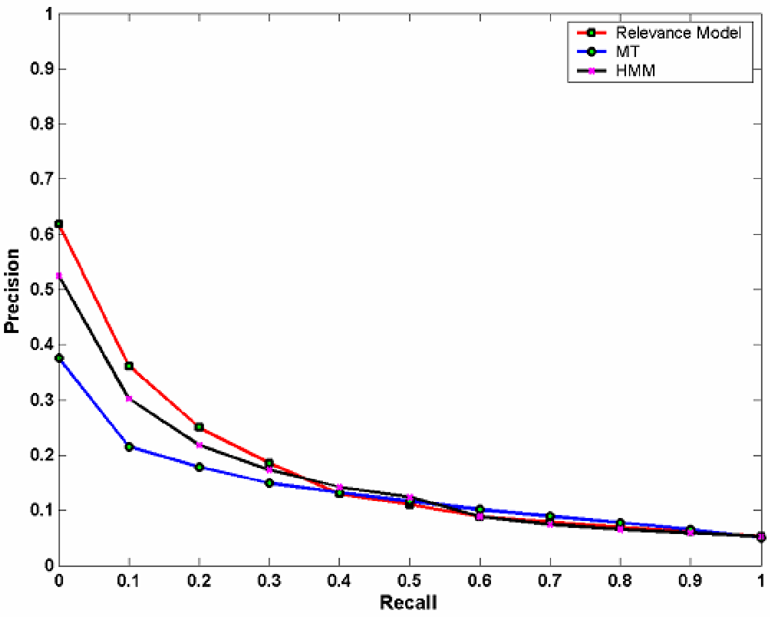


Figure 18: *Comparison of the HMM models with Relevance Models and Translation Models for image annotation*

	Chance	LM	SVM	Naive Bayes	Max Ent
mAP	0.05	0.125	0.116	0.102	0.1

Table 10: *Comparison of the different methods to extract concepts from ASR*

We also investigated mutual information as a means to extract features. In Fig. 23 we show the mAP for the most frequent five concepts for an increasing number of features. We can see that the optimal number of features depends on the concepts. Most striking is the difference between `indoors` and `outdoors`. `indoors` needs a very small number of features with an optimum below 1000 features, whereas for `outdoors` ten times as many features are necessary. However, the overall influence of feature selection is negligible.

8 Information Retrieval Experiments

8.1 Setup

8.2 Experiments

In Fig. 24 we compare linear with log-linear interpolation as a method to combine the different models. Here a combination of the baseline model with the machine translation model for concept annotation is done first. The difference between the two methods is surprisingly large. Usually in language modeling, we observe that log-linear interpolation is better than linear interpolation but the difference is never as large as in this figure.

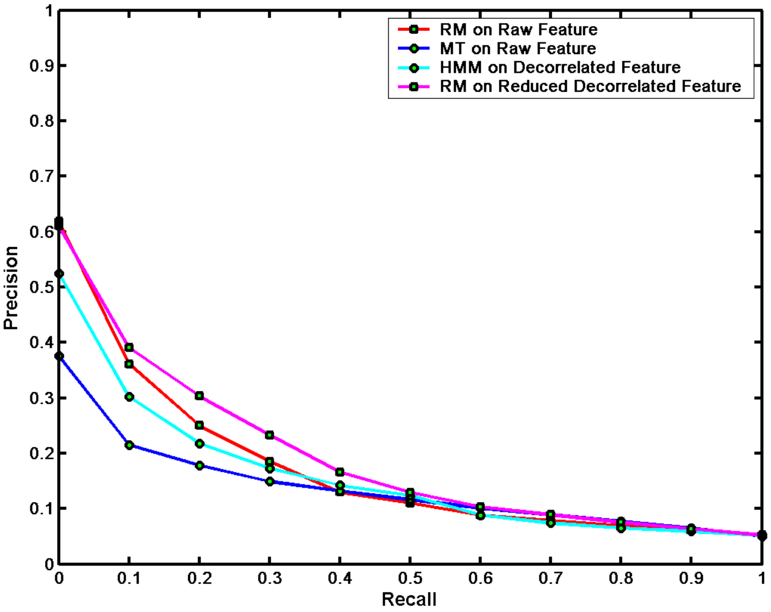


Figure 19: *Recall Precision graph demonstrating the improvements to the Relevance Models using the decorrelated, variance normalized feature set*

It is striking, that there is no interpolation weight where linear interpolation gives a benefit. This may be due to the fact that we have a problem in converting our concept annotation models into proper retrieval probabilities with a reasonable distribution of the probabilities in the interval [0 : 1].

An indication in the same direction is the fact, that the HMMs gave a comparable improvement but the optimal weights of the combination where in a completely different range.

In Tab. 11 we give the results of the fusion experiments. First the models extracting concepts from images where added. It turned out, that we could not turn optimal performance in concept annotation into good performance in retrieval. Instead in the combination experiment, the machine translation approach turned out to give best performance in combination. By throwing in the concept annotation models from ASR, an additional improvement was achieved.

Model	Retrieval mAP
Baseline	0.131
+ MT	0.139
+ Concepts from ASR	0.149

Table 11: *Results from fusing the different models*

Finally Fig. 25 gives the recall-precision curve of the overall best model, a combination of the baseline with the machine translation model for image annotation and the model that extracts concepts from ASR. We observe that we get a consistent improvement in the high-precision region.

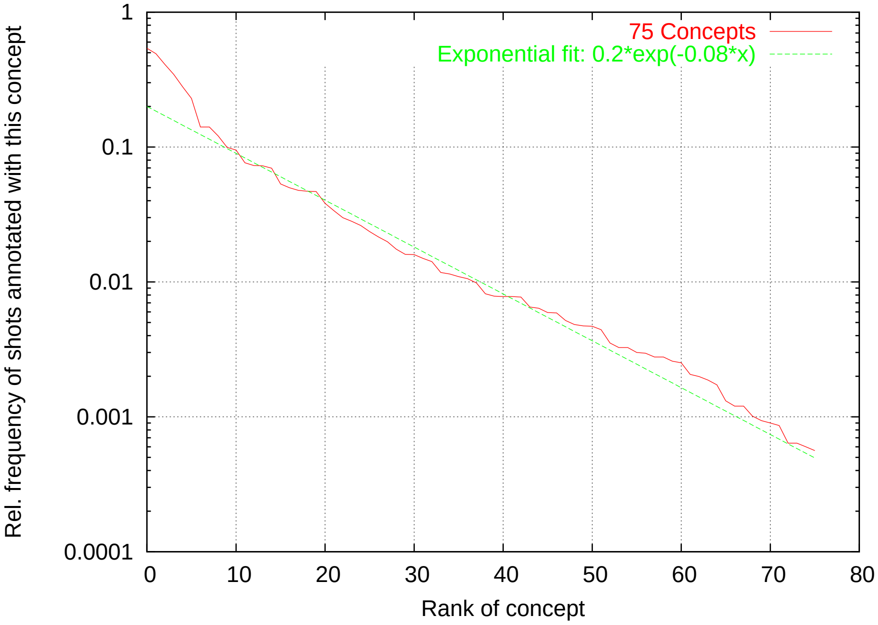


Figure 20: *Relative frequency of the 75 concepts sorted by their frequency*

9 Summary and Future Work

In this workshop, we investigated a novel approach for multimedia retrieval which jointly models the visual and textual components of a video shot. We built automatic multimedia retrieval systems using this approach. Experiments were conducted on the TRECVID03 corpus and initial results indicate that we get a 14% improvement in retrieval performance using joint models over a text-only baseline.

In particular, we investigated three distinct approaches for relating the visual part of the document to the text part of the query, namely Machine Translation, Relevance Models and Graphical Models. All three approaches were modeled as an information-bottleneck approach. We find that the Relevance Models provide the best performance compared with MT models and Graphical Models. For MT models, direct translation approach works best. HMM based approach that we investigated was started at the workshop and in the short duration, this approach emerged competitive to the more established approaches of Maching Translation models and Relevance models. To relate the visual part of the query to the ASR text of the video shot, we investigated several approaches for extracting visual concepts from ASR text, including MaxEnt models, Naive Bayes models and unigram count based models. These approaches indicate that predicting visual concepts from ASR, while a challenging and counter-intuitive task, does appear possible and perhaps even competitive to visual-only approaches. However, it is not clear what is the upper-limit on performance of such an approach.

Some of the challenges that we faced at the workshop included incomplete labeling of images (i.e. only a few concepts were marked in the images and not all the ones that were present). Also, these annotations were conducted by a large group of people (see NIST TRECVID common annotation forum) and the quality varied significantly between annotators. We did not exploit any spatial or temporal dependencies in our experiments. This needs to be better explored in future work. Also, expanding the size of the bottleneck and perhaps direct modeling of queries and documents needs to be explored. In our experiments, very little

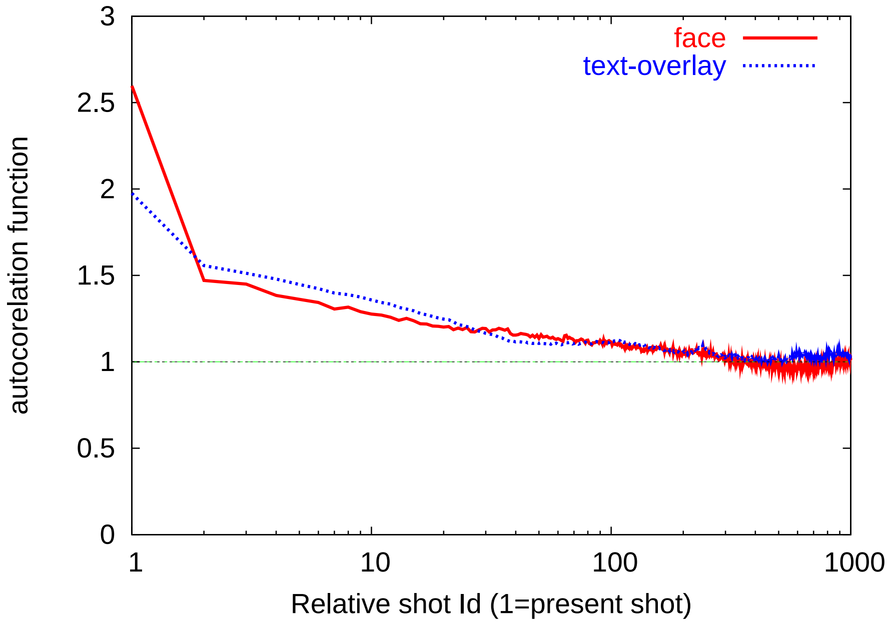


Figure 21: *Autocorrelation function for* Face *and* Text-Overlay *concepts*

query dependent processing was attempted. We note from literature that such techniques have worked well for several IR tasks. This is an important future direction for further performance improvements. One of the streams of information that we did not exploit in these experiments include on-screen text. Our assessment indicated that this information is very relevant for many queries. However, off-the-shelf OCR programs perform poorly on such images and produce significantly degraded text. If the quality of video OCR output can be improved, this source of information will become quite useful and can be easily integrated into the approaches that we developed here.

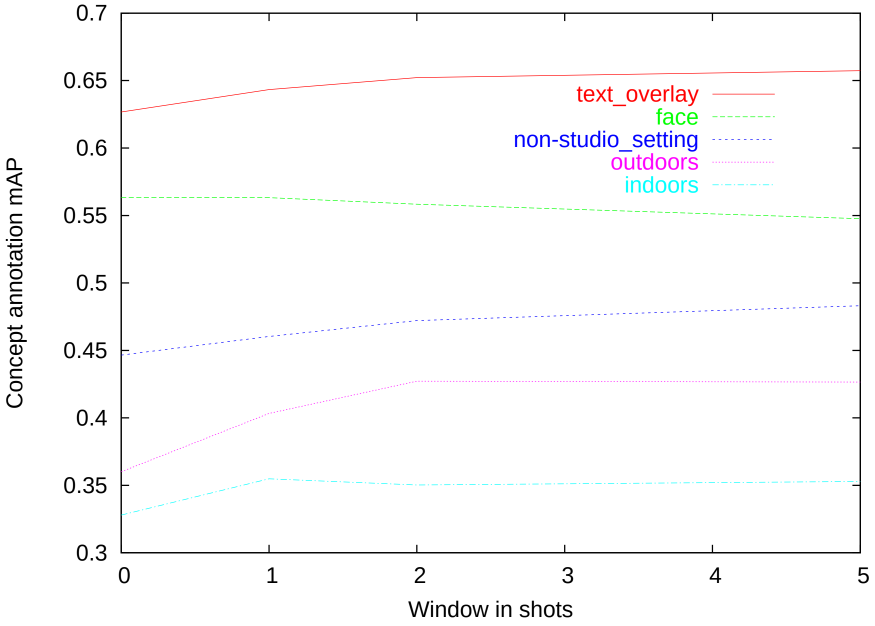


Figure 22: *mAP for the five most frequent concepts and various sizes of windows*

A Detailed DBN Models

A.1 Unconnected

Figure 26 shows the actual structure used to construct the unconnected model.

$$p(c_s) = \frac{1}{|C|} \tag{51}$$

$$p(c_0|c_s) = \frac{1}{|C|} \tag{52}$$

$$p(c_t|c_{t-1}) = \frac{1}{|C|} \tag{53}$$

$$p(f_t = 1) = \{ \begin{array}{c} 1 \text{ if } c_t \in Annotations \\ 0 \text{ otherwise} \end{array} \tag{54}$$

$$p(wt = 1) = 1 \tag{55}$$

$$\tag{56}$$

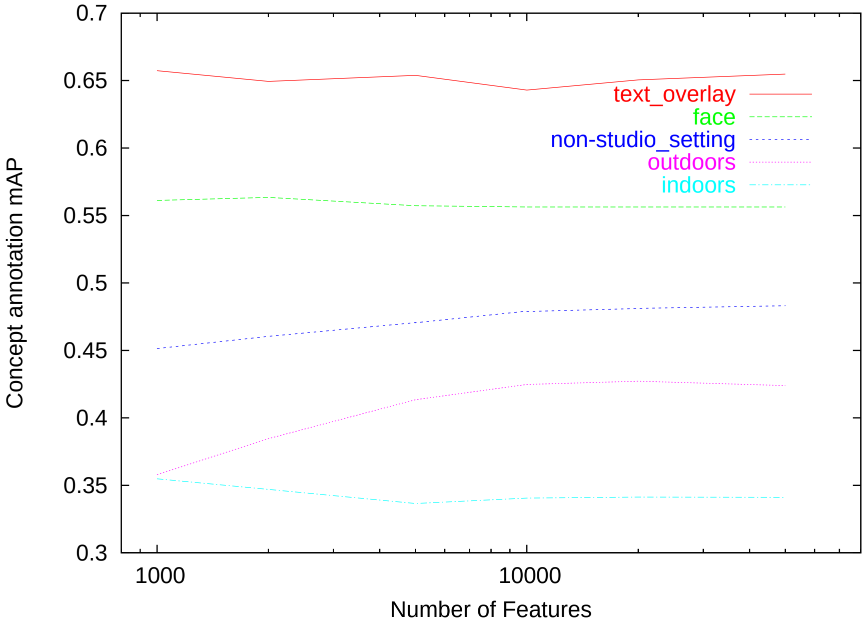


Figure 23: *Feature selection for the five most frequent concepts*

A.2 Chain

Figure 26 shows the actual structure used to construct the chain model. It is the same structure as the unconnected model; what differs is what the probabilities are.

$$p(c_s) = \text{empirical unigram probabilities of } c_s \tag{57}$$

$$p(c_0|c_s) = \text{empirical bigram probabilities of } c_t|c_{t-1} \tag{58}$$

$$p(c_t|c_{t-1}) = \text{empirical bigram probabilities of } c_t|c_{t-1} \tag{59}$$

$$p(f_t = 1) = \{ \begin{array}{c} 1 \text{ if } c_t \in \mathit{Annotations} \\ 0 \text{ otherwise} \end{array} \tag{60}$$

$$p(wt = 1) = 1 \tag{61}$$

$$\tag{62}$$

A.3 Grid

Figure 27 shows the structure of the grid connected model. This model captures horizontal and vertical dependencies using left-connected and top-connected nodes. Compared with the previous chain model, the state space goes as N^3 (as opposed to N^2).

References

[1] NIST, *TREC Video Retrieval Evaluation Conference(TRECVID2003)*, Gaithersburg, MD, November 2003.

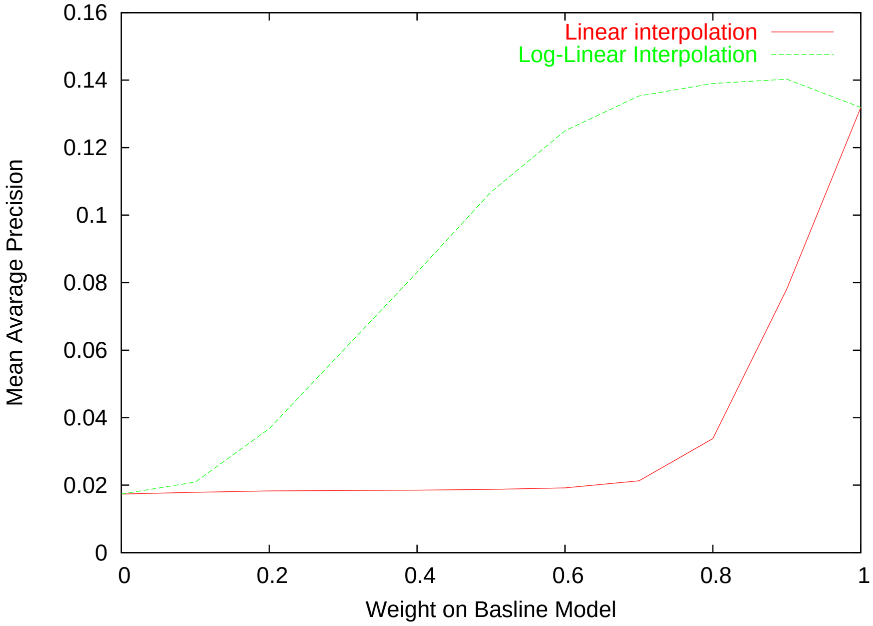


Figure 24: *Comparison of the two methods to combine models*

[2] J. M. Ponte and W. B. Croft, "A Language Modeling Approach to Information Retrieval," in *Proceedings of the Twenty-First Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*, Melbourne, Australia, 1998, pp. 275–281, ACM Press.

[3] John Lafferty and Chengxiang Zhang, "Document language models, query models, and rish minimization for information retrieval," in *Proceedings of the Twenty-Fourth Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*, New Orleans, Louisiana, 2001, pp. 111–119.

[4] Adam Berger and John Lafferty, "The Weaver System for Document Retrieval," in *Proceedings of the Eighth Text REtrieval Conference (TREC-8)*. 2000, pp. 163–174, NIST Special Publication 500-246.

[5] J. Jeon, V. Lavrenko, and R. Manmatha, "Automatic image annotation and retrieval using cross-media relevance models," in *Proceedings of the Twenty-Sixth Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*, Toronto, Canada, 2003, ACM Press.

[6] D. E. Maroulis, D. K. Iakovidis, S. A. Karkanis, and D. A. Karras, "CoLD: a versatile detection system for colorectal lesions in endoscopy video-frames," *(article in press) Computer Methods and Programs in Biomedicine*, 2004.

[7] P. Brown, S. Della Pietra, V. Della Pietra, and R. Mercer, "The mathematics of statistical machine translation: Parameter estimation," *Linguistics*, vol. 19, no. 2, pp. 263–311, 1993.

[8] P. Duygulu, K. Barnard, N. de Frietas, and D. A. Forsyth, "Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary," *Lecture Notes in Computer Science*, vol. 2353, pp. 97, 2002.

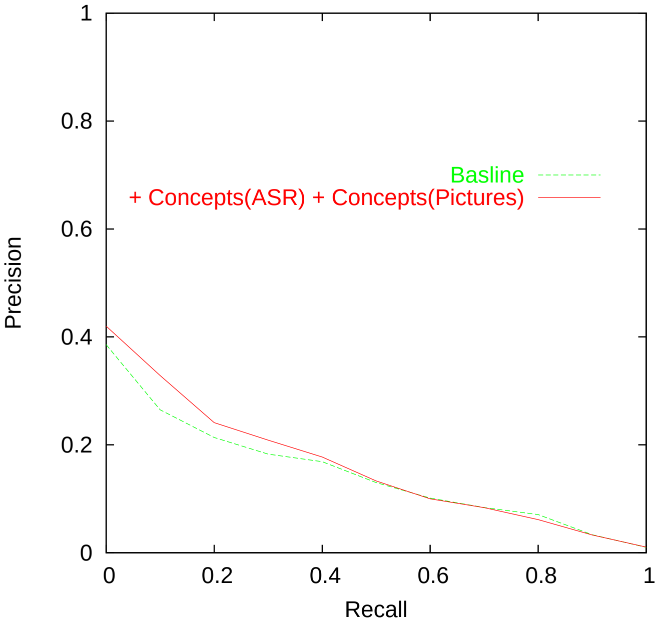


Figure 25: *Recall-Precision comparison between baseline system and the final fusion system*

[9] V. Lavrenko, M. Choquette, and W. B. Croft., "Cross-lingual relevance models," in *Proceedings of the 25th Intl. Conf. ACM SIGIR*, 2002, pp. 175–182.

[10] V. Lavrenko, R. Manmatha, and J. Jeon, "A model for learning the semantics of pictures," in *Proceedings of NIPS 2003*, 2004.

[11] S. L. Feng, R. Manmatha, and V.Lavrenko, "Multiple bernoulli relevance models for image and video annotation," in *Intl. Conf. on Computer Vision and Pattern Recognition*, Washington D.C., June 2004.

[12] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE T. Patt. Analy. and Mach. Intell.*, vol. 22, no. 8, pp. 888–905, 2000.

[13] V. Lavrenko, S.L.Feng, and R. Manmatha, "Intl. conf. on acoust., sp., and sig. proc.," Montreal, QC, May 2004.

[14] R. Manmatha, S.L. Feng, and V. Lavrenko, "Associating words and pictures," *Artificial Intelligence Journal*, (under submission).

[15] S. Young, D. Kershaw, J. Odell, D. Ollason, V. Valtchev, and P. Woodland, *The HTK Book*, Entropic, Cambridge, 2000.

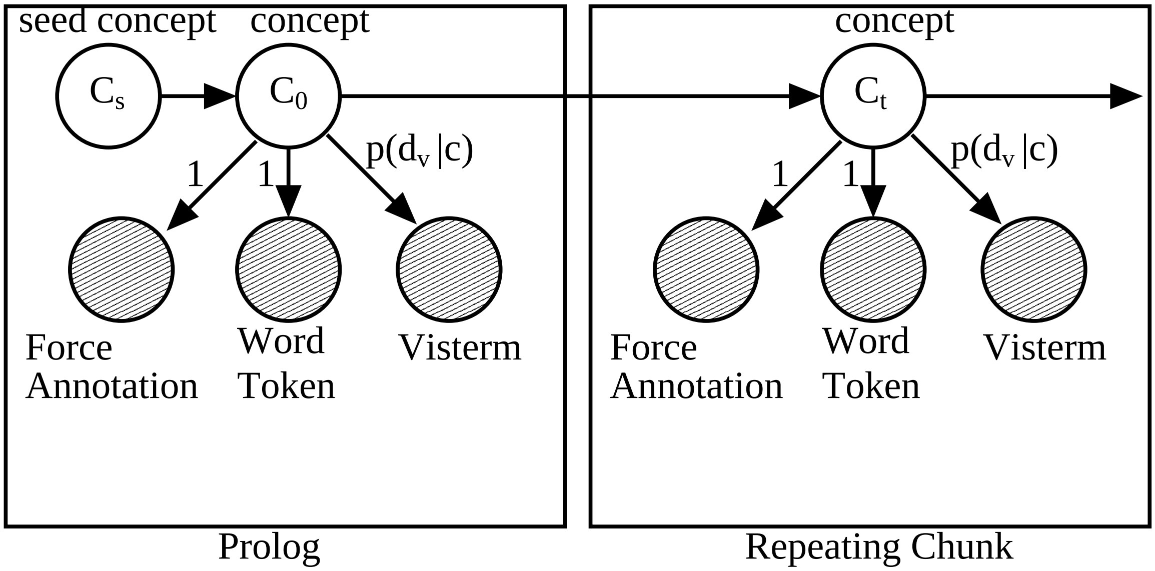


Figure 26: Unconnected Model and Chain Model

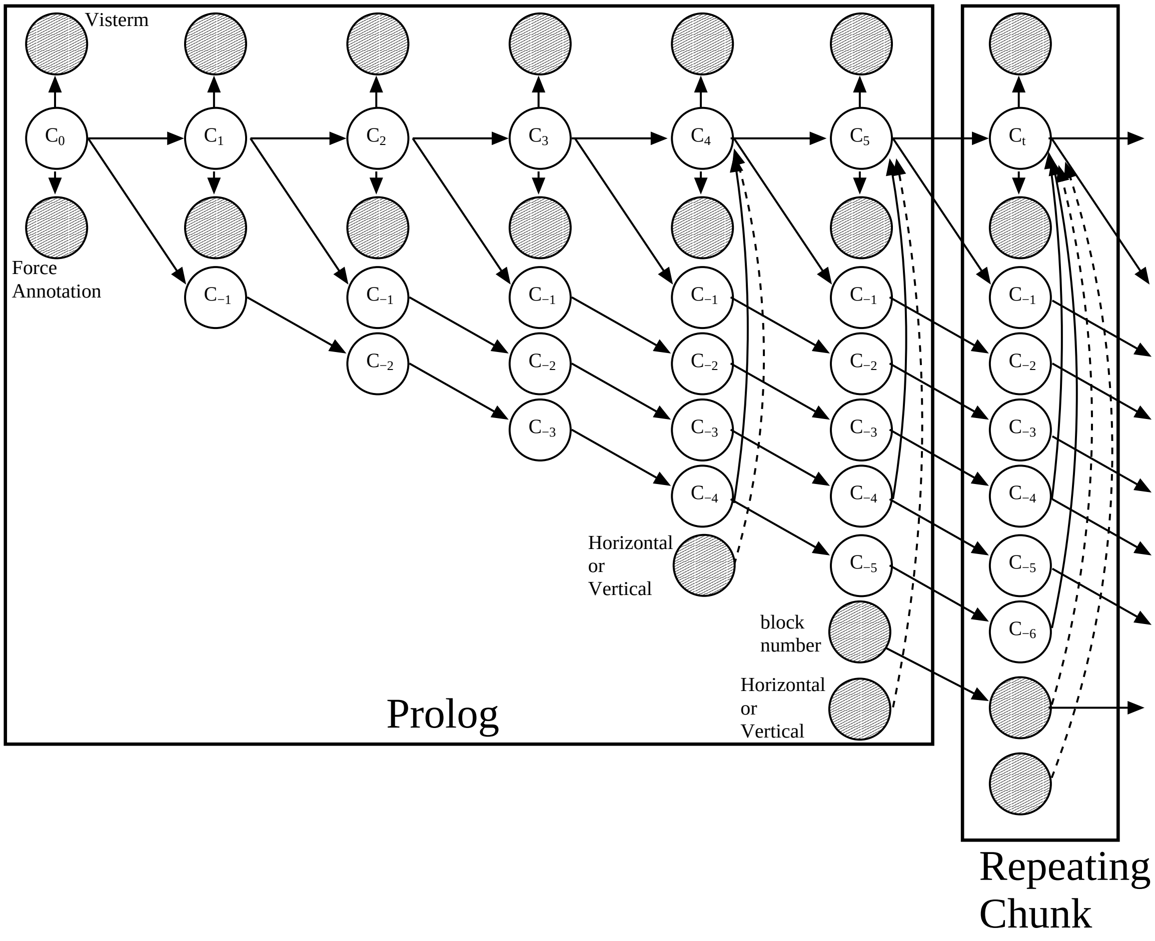


Figure 27: Full Dependence Model