Fast Unsupervised Alignment of Video and Text for Indexing/Names and Faces

Subhransu Maji
University of California, Berkeley
California, USA
smaji@cs.berkeley.edu

Ruzena Bajcsy
University of California, Berkeley
California, USA
bajcsy@cs.berkeley.edu

ABSTRACT

We propose a novel way of combining weakly associated video/audio and text steams in an unsupervised manner which is faster than conventional speech recognition. The technique aligns audio/video and text streams which will enable video search using the associated text. Multimedia of this form includes news broadcast with summaries, parliament proceedings and court trials with transcripts, sports telecast with text commentary, etc. We also show how we can annotate the video with the names of the person appearing in the video which will allow name based indexing/search. We test the technique on a 80 minute video segment downloaded from the website of the International Court of the Former Yugoslavia, with the corresponding transcripts. The proposed technique achieves 88.49% accuracy on sentence level alignments and 95.5% accuracy on the task of assigning names to faces.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Clustering; I.7 [**Document and Text Processing**]: Miscellaneous

General Terms

Algorithms

Keywords

Multimedia alignment, Indexing, Names and Faces

1. INTRODUCTION

The problem of combining various streams of information coherently for various tasks like search, organization for better browsing, is quite challenging. Large amounts of data exits on the web which contain audio, visual and text information together. These can be divided into two categories, one which has the various forms of media synchronized with one another and the other which do not. Examples of the former include photos with captions, movies with subtitles, etc. The later include news broadcast with the text taken from the newspapers of the same story, proceedings of parliament/courts with transcripts, etc. One can obtain these complimentary sources of information from the web for example by a video, image and text search on the same item. Clearly there is a huge amount of data on the web for which only weak associations exist and any method to organize them is useful. One such source of data are the world court tribunals. These tribunals have been created to preserve, archive the legacy and organize the data in such a way that it enables students of law, future lawyers, etc, to easily find any information they need.

In this work we show a technique of obtaining alignments of video and text, in the setting of court room style proceedings with transcripts. The alignment can then be used to annotate, label the video for various tasks. As an example we show how to automatically annotate the faces appearing in the video with the names using the transcripts.

Problem Challenges: The transcripts are noisy as they are taken down manually and are edited to remove sensitive information. Also in many of the trials, especially in the international courts, there is a translator who translates from the native language to English. This induces a lag and lot of additional noise which are not present in the transcripts, rendering the task of alignment non trivial. The whole area of face recognition itself is hard because of the variation in pose, expressions etc. The overall task is as described in the Figure 1. The two main tasks are

1. Computing the alignment between the Video/Audio and the text at the sentence level.

2. Using the alignment to obtain a correspondence between the names and faces in the video.

The rest of the document is structured as follows; Section 2 describes the previous work done in this area. In section 3 we describe the process of computing the sentence level alignments and evaluate various techniques. Section 4 contains the details of face detection, and section 5 deals with the problem of assigning names to faces and comparison of various algorithms. We conclude and provide directions for future research in section 6. Section 7 contains acknowledgments.

2. PREVIOUS WORK

A huge amount of literature exists on combing visual and the text steam together for the task of image understanding

MS'07, September 28, 2007, Augsburg, Bavaria, Germany.
Copyright 2007 ACM 978-1-59593-782-7/07/0009 ...$5.00.

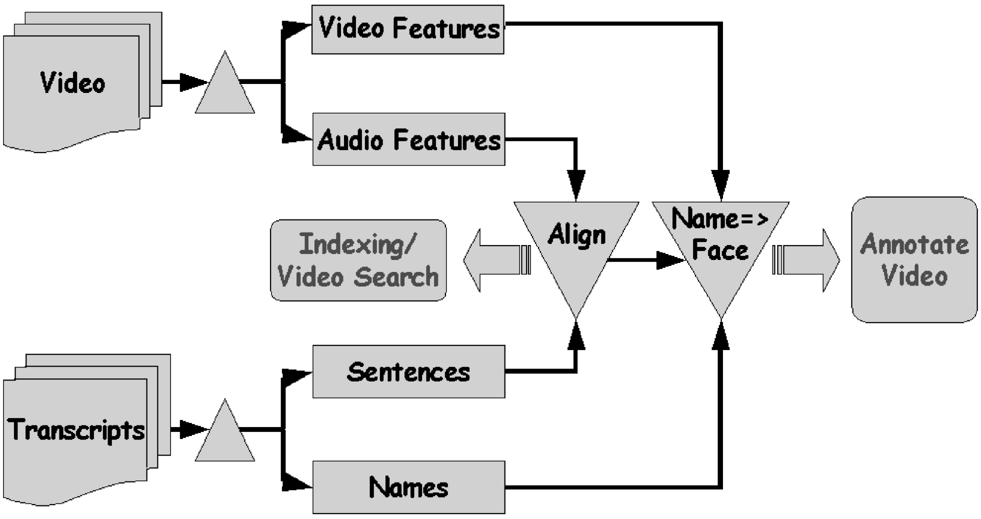

Figure 1: Flowchart describing the steps of the algorithm.

and search. Tamara Berg *et al* [3] show how to construct a face dataset from a collection of automatically gathered news photographs and captions. The general task of attaching keywords to images itself has has received considerable attention in [9, 8, 19]. These methods use variations of multiple instance learning which is a way to build classifiers from bags of labeled examples. Belongie *et al* [4] demonstrate examples of joint image-keyword searches. Barnard and Johnson [2] show one can disambiguate the senses of annotating words using the images. The central theme of all these works is that text and images contain complimentary information and one can combine them together to solve problems which are otherwise hard to solve by themselves.

In the domain of video, Aranjelovic and Zisserman [1] , show how to do automatic face recognition in feature length films. Their system allows one to search for all occurrences of frontal faces in the movie given a small set of query images. Extending the detection to profile and three-quarter views, K.Mikolajczyk *et al* [10] give a temporal approach to reliably detect frontal and profile faces in a video. They use zero order dynamic model for appearance variation and use condensation filter to accumulate probabilities of face detection over time. The Condensation algorithm (Conditional Density Propagation) proposed by Isard and Blake [7] allows quite general representations of probability and the use of non-linear motion models more complex than those commonly used in Kalman Filters. Mark Everingham *et al* [5] show how to automatically name the faces in the video using the transcripts. This comes closest to the current work, but in their case they have subtitles which they use in conjunction with the transcripts to find an alignment between the video and the transcript. In our case we only have the transcripts and finding the alignment itself is a challenging task.

In the speech recognition community a number of researchers have examined a variety of ways to handle quickly transcribed data. We follow the methodology similar in spirit to Anand *et al* [16], which describes an efficient repair procedure for quickly taken down transcripts. The focus there was to compute word level alignments of audio segments which can then be used as training data. The step however required manual alignment of the audio files to a set of transcripts which can be time consuming. The proposed method does not require this step. Yet another way to compute alignments between the text and the video would be to run a speech recognizer to obtain the speech (and hence the subtitles) which can then be used to align it with the transcript. However conventional ASR for large vocabulary is slow. As a comparison, on our dictionary which is of the order of HUB4 dataset, the CMU's SPHINX-4 speech recognition system, has a realtime rate of 3.95 on a dual CPU UltraSPARC(R)-III running at 1015 MHz with 2G of memory [14]. Compared to this the proposed method takes about an hour for a 80 min video on a window of 11 sentences on my laptop (IBM ThinkPad, Centrino duo, 1 GB of RAM), realtime factor of 60/80 = .75.

3. ALIGNING SENTENCES

In this section we describe the process of obtaining sentence level alignments from a video and the corresponding transcript. For the task of search, word level alignments are perhaps less useful and we model this problem as a sentence alignment problem, though for applications like speech recognition where alignments could be used a training data, word level alignment(or even finer) are necessary. We use the word sentence loosely as defined by a sequence of W words and the corresponding entity in the video to be T length segment.

3.1 Obtaining the Video

he video files are obtained by dumping the streams available for public viewing from the International Criminal Tribunal for the former Yugoslavia's (ICTY) website. A 80 minute segment with the corresponding transcript was downloaded from the website which serves as test data for the task. The video is low resolution (256x192) at 15 FPS. Figure 2 shows a few frames from the video. The video contains almost no clutter and has most of the faces in frontal poses, which is typically the case with video of this kind.

Figure 2: Sample frames from the video.

3.2 Audio Feature Extraction

The audio stream is converted into a RIFF (little-endian) data, WAVE audio, Microsoft PCM, 16 bit, mono 16000 Hz. format and broken down into non overlapping 15 second fragments, using opensource mplayer [11] and SOund Exchange (sox) [13]. The front-end transforms these waveforms into a sequence of 39-dimensional PLP features (12 mel-frequency cepstral parameters plus energy, and 1st/2nd derivatives), with frames representing 25ms windows at 10ms intervals. Side-based mean and variance normalization were applied, although without vocal tract length normalization. All the feature extraction, processing, alignments, etc were done using the Hidden Markov Model Toolkit popularly know as HTK [6]. HTK is a portable toolkit for building and manipulating hidden Markov models and has built in tools for speech recognition.

3.3 Text Processing

The transcripts that are available for download are in html format and we process it to remove all the html tags, punctuation marks and other formatting information. The speaker information is extracted from each of the lines and few other manual preprocessing is applied to convert the data into sequence of words and are converted to HTK's standard MLF format. For this task we also remove the utterances of numbers and dates, though one could transcribe them. These are then broken into files containing 20 words each. The number 20 was chosen so that the number of audio files and the text files are of roughly the same, and is about the length of a sentence. However, no care is taken to chop the lines on sentence boundaries, and each 20 word sequence could contain words from multiple sentences and could be spoken by multiple speakers. A section of the transcript is shown Figure 3.

3.4 Computing Local Alignment Scores

The alignment score of an audio file and a sequence of words (henceforth referred to as a sentence) is done as follows: First we create a dictionary containing just the words from the sentence and their phonetic pronunciations, using a full dictionary of 121942 English words (with multiple ways to pronounce a single word). We use a monophone model with 12 gaussian mixtures per phone trained on 265 hours of American English CTS(Conversational Telephone Speech). From each of the sentences a lattice is created which represents the transition and emission probabilities of the words in the sentence. Between each word we add a filler model, which can model arbitrary speech, to allow for words which are spoken, but not in the transcript. This might have resulted because of errors in transcription, later editing of the sentences to remove sensitive information, pauses in speech, or words spoken in a different language, which results in the transcript being non verbatim. Next, we align the audio to the sentence by the Viterbi algorithm using the HTK's implementation called HVite and compute the log-likelihood of the audio given the sentence. This process is fast due to the restricted vocabulary of 20 words and takes a couple of seconds for aligning the 15 second audio with the sentence. Figure 7 illustrates the alignments.

3.5 Computing Global Alignments

The rough correspondence can be obtained by a linear interpolation, i.e. the sentence number is proportional to the audio file number. This is the baseline algorithm for this task. However the true sentence is likely to be somewhere around the target so we perform a window search near the it. So we compute the local alignment scores for all the sentences within the window as described in the previous step.

There are various ways to combine all the scores to obtain a final alignment. In this work we try the following three schemes:

- **Global Dynamic Programming:** The global alignment between all the audio and the text at a sentence level can be computed using a version of string edit distance (dynamic-time-warping), for which we have efficient dynamic programs. The cost of matching $audio_i$ and $sentence_j$ is the log-likelihood of the $audio_i$ under the HMM obtained from $sentence_j$.

- **Best Line:** An alternate approach to just select the sentence, within a window of W sentences on either side of the target predicted by linear interpolation, which assigns the highest log-likelihood for the the audio. Unlike the previous method this does not give a monotonic assignment.

- **Local Dynamic Programming:** This is a modified version of local string alignment algorithm, which seeks to find substrings of two strings which align well with each other. We modify it as follows; First it locates local maxima of log likelihood in sentence and audio segment space, which are separated by at least stepsize ± window-size. These are anchor points and the global dynamic programming is constrained to pass through these points. This can be done efficiently by independently computing a string alignment between successive anchor points. The step-size and window-size are tunable parameters and we discuss its effect in the next section.

3.6 Alignment Evaluation/Results

Manually obtained alignment for each 15 minute audio segment serves as the ground truth. Annotating 250 segments with the sentences it aligns to, takes about 3 hours manually. Since the boundaries do not match a single sentence could be in multiple audio segments and similarly a audio file could align to multiple sentences. The score of an alignment is the fraction of the predicted sentences which lie within the correct lines(inclusive) for all sentences.

Figure 5 gives the sentence alignment accuracies of various techniques. The baseline algorithm for this task, as we described earlier, is the one which simply predicts the sentence using linear interpolation. This performs poorly with

```
JUDGE ROBINSON: I'm not hearing -- I'm not hearing any comments on that. I'm not receiving any comments on that. If that is what you have to say, we'll call the witness.
Please call the witness.
THE ACCUSED: [Interpretation] Very well, but I want it to remain on the record that I demand that the Appeals Chamber ...
[The witness entered court]
WITNESS: JAMES BISSETT [Resumed]
[Trial Chamber confers]
JUDGE ROBINSON: Mr. Milosevic, proceed with the examination-in-chief.
THE ACCUSED: [Interpretation] Thank you, Mr. Robinson.
```

Figure 3: A section of the transcript. The set of speakers extracted from the entire transcript are: *judge bonomy, judge kyon, judge robinson, mr. milosevic, mr. nice, the accused, the interpreter, the witness.*

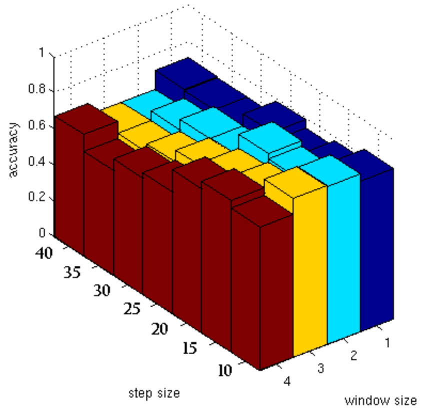

Figure 4: Plot alignment accuracy as a function of the step size and window size. A step size of 10 with with a window size of 3 works best for this dataset giving an accuracy of 88.49%.

Method	Accuracy
Baseline	13.01 %
Global DP	66.50 %
BestLine(w=15)	76.96 %
Local DP	88.49 %

Figure 5: Sentence alignment accuracy of various techniques.

only 13.07% of the predictions falling in the right positions. The audio contains many silence regions and varying speed of speech which causes the baseline algorithm to perform rather poorly.

The best line approach with a window of 15 around the target , gives an accuracy of 76.96%. The global dynamic programming algorithm actually makes the accuracies worse to 66.5%, as it favors more of a diagonal path, as it is shorter. Also the cost (log -likelihood) of aligning an audio segment to an arbitrary text is perhaps not very meaningful as the effect of the filler model takes over if none of the spoken words are in the sentence. In contrast the local string alignment gives an accuracy of 88.49% for a step-size of 10 and a window-size of 3. Figure 6 shows local alignment scores and the predictions of various algorithms and Figure 4 shows the alignment accuracies as a function of the step-size and the window-size.

4. VIDEO PROCESSING

The video stream is converted into a sequence of images by sampling 1 frame every second from the video. These images are then used to extract faces which appear in the video. Face detection has been a active area of research and many successful techniques exist in the literature. We refer interested readers to http://www.facedetection.com/ which is a repository of papers, datasets, software, etc related to face detection. We use the open source face detector which is an implementation of the face detection algorithm proposed by Viola and Jones [17] in opencv-1.0.0 [12]. In the video we find that most of the faces in the video are frontal and almost all of them are detected. Profile face detection remains a challenging task and in this work we do not detect profile faces, though integrating a profile detection into the framework in straightforward. The detected faces along with a 10%, neighborhood around it is then normalized into a $50x50$ grayscale patch.

5. ASSIGNING NAMES TO FACES

The task is to assign a name to each face that is detected in the previous step. For each $\triangle T$ seconds interval (15 seconds, here) we extract faces and the corresponding names of the people who spoke the sentences which were aligned to it. Multiple faces and names could occur as we do not take special care of segmenting the video and text into sentence boundaries. Though it is likely that the person who is speaking is the person who appears in video, it is not always true, which makes this task non trivial. We use two different techniques for comparison described in the next two subsections. The ground truth is obtained by manually assigning each of the faces the correct name. Figure 8 gives the name to face assignment accuracies of various techniques.

5.1 Baseline - Voting

The simplest scheme would be to assign a random name in the set of persons who spoke the sentences, achieves a mean accuracy rate of 63.19% over 50 trials (median 63.7 %). Though the baseline algorithm performs rather poorly,

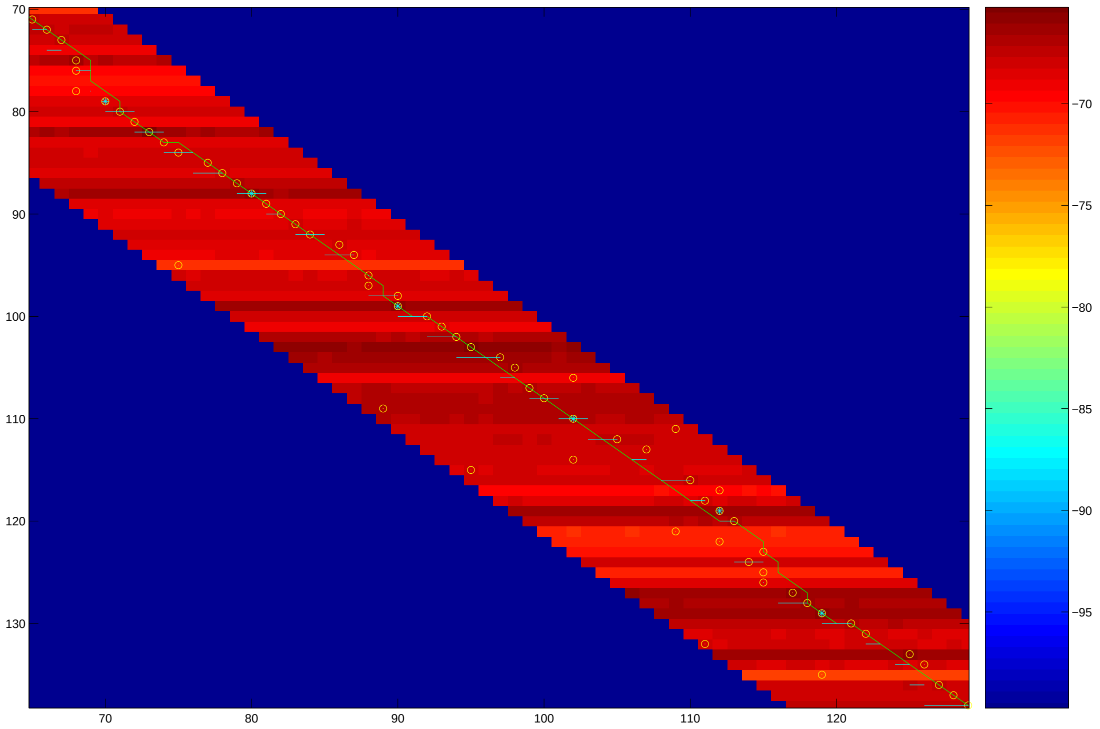

Figure 6: A section of the alignment scores array(X-axis Sentences, Y-Axis Audio). The values are drawn in jet scale with blue being the least and dark red being the highest. The yellow circular dots are the best sentence for each line. The magenta horizontal bars at even audio coordinates, are the ground truth and the green line passing through the lattice near diagonally is the path computed by the Local Dynamic Programming. Magenta Dots are the anchor points.

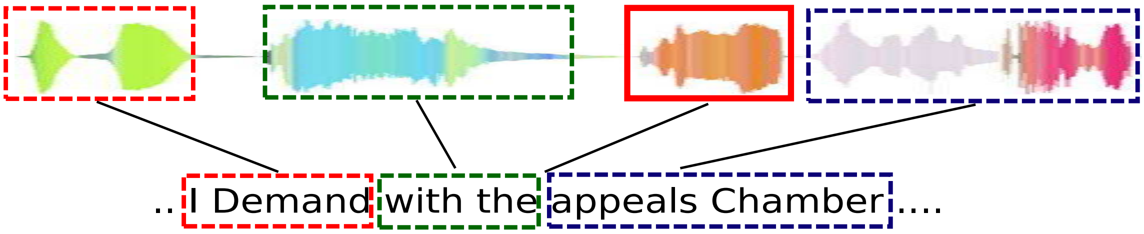

Figure 7: A illustration of the alignment. The portion of the waveform which is not in the transcript (3rd box from left on the top row) is matched to the filler model between words.

Method	Accuracy
Baseline W.O Alignment	12.5 %
Baseline With Alignment	73.2%
Cluster Vote	95.5 %

Figure 8: Accuracy of assigning names to faces of various techniques.

but it shows that there is a high correlation between who is speaking and who is actually in the frame. In contrast without the alignment, assigning each face image to one of the random names, would give an accuracy of 12.5% (8 names).

5.2 Cluster Vote

An improved scheme would be to first cluster the faces and let each face vote for the names in the sentences it is aligned to and the cluster is assigned the majority name. To do this firstly, each $50x50$ face detected in the face-detection step, is gaussian blurred using a $2x2$ kernel to remove any aliasing effects and compression artifacts. We then concatenate the grayscale values to a 2500 length feature vector, which is the representation for our faces. We then use the K-Means with L2-distance for clustering the vectors.

We randomly select about 20 segments for test. A typical result of clustering and voting is shown in Figure 9. Note that only 4 names appear in these segments so we show the votes for only these. The recognition accuracy is about 84.84% (95% on correct faces), averaged over # clusters $\in \{4, 5, 6, \ldots, 12\}$ and 10 random initializations per choice of # clusters. We find that the accuracy is fairly invariant to the number of clusters as long as the clusters is more than the number of faces as shown in Figure 10. Also about 20 out of the 225 images of faces are false detections and 2 faces appear which do not have the corresponding name in the text, which cannot be correctly classified, so the best we can get is to about 90.22%. Most of the legitimate errors from faces which appear rarely or always with some other faces in the time quanta we break up the audio/text segments into and since the votes are local, the other faces get the same number of votes or possibly more. Clusters of false face detections also arise which should ideally be not assigned to any face, but our voting scheme does not take into the account the fact that the faces assigned a name can only be projections of an underlying 3D face. The rotation could be either in plane or out of plane. One way to correct for this is to rectify the faces to a canonical pose by detecting facial features like corner of eyes,nose and rectifying the face. However given the low resolution of the video and even smaller faces, it is unlikely that that these features can be reliably extracted, and in this work we do not explore this technique.

6. SUMMARY AND CONCLUSIONS

We propose a novel way of aligning the audio and text streams, which is faster than conventional ASR and show that it can be used to annotate individuals appearing in the video, with their names. The alignment can also be used to index the video, making it easily browsable/searchable. Given the large amount of asynchronous video and text available on the world wide web and the lack of proper ways to search it, this is one way one might go on to organize the data. The technique is quite general and no modeling decisions specific to the data were made, though in one should test the technique on more complex data, especially where multiple people appear together and speak short sentences after one another. Finally, the accuracies for face detection can be improved if instead of looking at every frame, we use temporal information to eliminate false positives and detect less probable faces, for e.g. like the technique suggested in [10] and accuracies for face recognition can be improved by considering smarter features, like eigenfaces [15] and kLDA-kPCA [18].

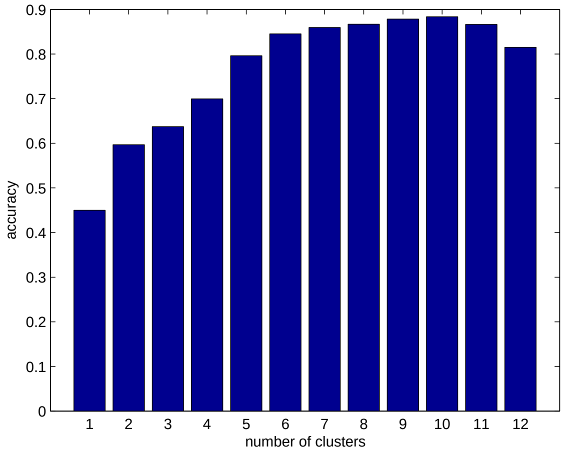

Figure 10: Accuracy of face labeling as a function of the number of clusters.

7. ACKNOWLEDGMENTS

I would like to thank Arlo Faria, for providing me with the acoustic models, dictionary, as well as helping me setting up HTK. Also thanks to Professor David Cohen and James Luong for pointing me to the datasets.

8. REFERENCES

[1] O. Arandjelovic and A. Zisserman. Automatic face recognition for film character retrieval in feature-length films. In *CVPR05*, pages I: 860–867, 2005.

[2] K. Barnard, M. Johnson, and D. Forsyth. Word sense disambiguation with pictures. In *Proceedings of the HLT-NAACL 2003 workshop on Learning word meaning from non-linguistic data*, pages 1–5, Morristown, NJ, USA, 2003. Association for Computational Linguistics.

[3] T. L. Berg, A. C. Berg, J. Edwards, M. Maire, R. White, Y. W. Teh, E. Learned-Miller, and D. A. Forsyth. Names and faces in the news. In *omputer Vision and Pattern Recognition*, pages 848–854, 2004.

[4] C. Carson, S. Belongie, H. Greenspan, and J. Malik. Blobworld: image segmentation using expectation-maximization and its application to image querying. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(8):1026–1038, 2002.

[5] M. Everingham, J. Sivic, and A. Zisserman. Hello! my name is... buffy – automatic naming of characters in

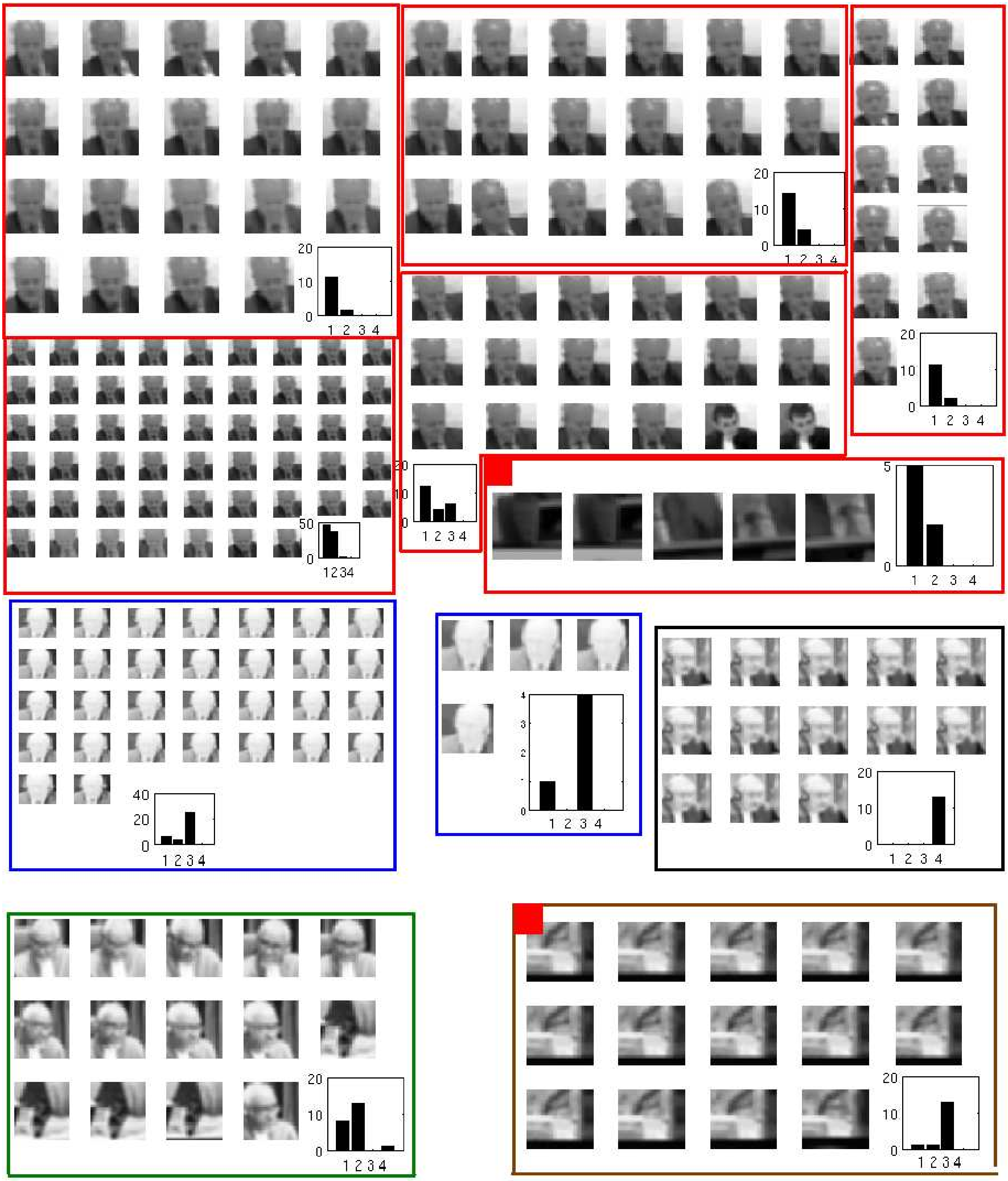

Figure 9: Various clusters with the distribution of votes for 4 names. There are a few clusters of false detections and some clusters have different faces.

tv video. In *Proceedings of the British Machine Vision Conference*, 2006.

[6] The hidden markov model toolkit (htk), machine intelligence laboratory, cambridge university engineering department,http://htk.eng.cam.ac.uk/.

[7] M. Isard and A. Blake. Condensation conditional density propagation forvisual tracking. *Int. J. Comput. Vision*, 29(1):5–28, 1998.

[8] O. Maron and T. Lozano-Pérez. A framework for multiple-instance learning. In M. I. Jordan, M. J. Kearns, and S. A. Solla, editors, *Advances in Neural Information Processing Systems*, volume 10. The MIT Press, 1998.

[9] O. Maron and A. L. Ratan. Multiple-instance learning for natural scene classification. In *Proc. 15th International Conf. on Machine Learning*, pages 341–349. Morgan Kaufmann, San Francisco, CA, 1998.

[10] K. Mikolajczyk, R. Choudhury, and C. Schmid. Face detection in a video sequence: A temporal approach. In *CVPR01*, pages II:96–101, 2001.

[11] Mplayer - the movie player, http://www.mplayerhq.hu/design7/info.html.

[12] Open computer vision library, http://sourceforge.net/projects/opencvlibrary/.

[13] Sox - sound exchange, http://sox.sourceforge.net/.

[14] Cmusphinx: The carnegie mellon sphinx project, http://cmusphinx.sourceforge.net/html/cmusphinx.php.

[15] M. Turk and A. Pentland. Face recognition using eigenfaces. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, page 586âĂŞ591, 1991.

[16] A. Venkataraman, A. Stolcke, W. Wang, D. Vergyri, V. R. R. Gadde, and J. Zheng. An efficient repair procedure for quick transcriptions. In *Proceedings of ICSLP*, 2000.

[17] P. Viola and M. Jones. Robust real-time object detection. *International Journal of Computer Vision - to appear*, 2002.

[18] M.-H. Yang. Kernel eigenfaces vs. kernel fisherfaces: Face recognition using kernel methods. In *FGR '02: Proceedings of the Fifth IEEE International Conference on Automatic Face and Gesture Recognition*, page 215, Washington, DC, USA, 2002. IEEE Computer Society.

[19] Q. Zhang and S. Goldman. Em-dd: An improved multiple-instance learning technique, 2001.