Narrative Theme Navigation for Sitcoms Supported by Fan-generated Scripts

Gerald Friedland
International Computer Science Institute
1947 Center Street, Suite 600
Berkeley, CA 94704-1198
fractor@icsi.berkeley.edu

Luke Gottlieb
International Computer Science Institute
1947 Center Street, Suite 600
Berkeley, CA 94704-1198
luke@icsi.berkeley.edu

Adam Janin
International Computer Science Institute
1947 Center Street, Suite 600
Berkeley, CA 94704-1198
janin@icsi.berkeley.edu

ABSTRACT

The following article presents a novel method to generate indexing information for the navigation of TV content and presents an implementation that extends the Joke-O-Mat sitcom navigation system, presented in [1]. The extended system enhances Joke-o-mat's capability to browse a sitcom by scene, punchline, dialog segment, and actor with word-level keyword search. The indexing is performed based on the alignment of the multimedia content with closed captions and "found" fan-generated scripts processed with speech and speaker recognition systems. This significantly reduces the amount of manual intervention required for training new episodes, and the final narrative-theme segmentation has proven indistinguishable from expert annotation. This article describes the new Joke-o-mat system, discusses problems with using fan-generated data, and presents results on episodes from the sitcom *Seinfeld*, showing segmentation accuracy and user satisfaction as determined by a human-subject study.

Categories and Subject Descriptors

H5.5 [**Information Interfaces and Presentation**]: Sound and Music Computing—*Signal analysis, synthesis, and processing*; H5.4 [**Information Systems Applications**]: Navigation

General Terms

semantic navigation

Keywords

acoustic video segmentation, narrative-themes, crowd sourcing, broadcast TV

1. INTRODUCTION

In the VCR era, content-based navigation was limited to play, pause, fast-forward, and fast-rewind. Thirty years later, videos are watched in many different ways: DVD and Blu-ray disc players, on-demand content, and sharing of home-made content over the Internet to name just a few. This, along with increasingly diversified channel options, has greatly increased the amount of multimedia data available. This plethora of content makes it increasingly difficult to find the specific information one desires. However, except for manually annotated chapter boundaries and other specialized scenarios, our ability to navigate multimedia content is much the same as in the era of the VCR. Fortunately, professionally produced TV and radio content usually contains acoustic markers which labels relevant portions of the content (e.g. music for scene transitions, or laughter for punchlines) that can be exploited for better navigation.

The following article presents a novel method to generate indexing information for the navigation of TV content and presents an implementation that extends the Joke-O-Mat sitcom navigation system, presented in [1, 2]. The extended system enhances Joke-o-mat's capability to browse a sitcom by scene, punchline, dialog segment, and actor with keyword search using the automatic alignment of the output of a speaker identification system and a speech recognizer with both closed captions and found fan-generated scripts. This method significantly reduces the amount of manual intervention required for training new episodes, and the final narrative-theme segmentation has proven indistinguishable from expert annotation (as determined by a human-subject study). The article is structured as follows: In Section 2 we discuss previous and related work. Section 3 introduces the use case and how the enhanced Joke-o-mat system applies to it. Section 4 presents a brief description of the system presented in [1, 2] and points out the limits of the original system which motivate the new approach presented in Section 5 and evaluated in Section 6. Section 7 presents the limits of the improved system along with future work. Section 8 concludes the article.

2. RELATED WORK

There is a wealth of related work in multimedia content analysis, especially broadcast video analysis, including [3, 4]. A comprehensive description of the related work would easily exceed the page limit for this article. Therefore, we survey only part of the most directly relevant work; see [7] for a more complete summary.

The TRECVid evaluation [6], organized on a year-by-year basis by the US National Institute of Standards and Technologies (NIST), investigates mostly visual event detection

AIEMPro'10, October 29, 2010, Firenze, Italy.

on broadcast videos [6]. The task is to detect concepts like "a person applauding" or "a person riding a bicycle". While many methods developed in the community are very interesting for the research presented here and should absolutely be used to complement and extend it, the TRECVid evaluation does not concentrate on navigation-specific events but rather on concept detection tasks. Its counterpart, the NIST Rich Transcription (RT) [5] evaluation, focuses on acoustic methods for transcribing multimedia content. The evaluation is currently focusing on meeting data but previous evaluations included broadcast news from radio and television.

The Informedia project's [8] basic goal is to "achieve machine understanding of video and film media, including all aspects of search, retrieval, visualization and summarization in both contemporaneous and archival content collections". The main focus is retrieval of videos from a large database. Navigation interfaces are not explored on the level proposed here.

Our approach, on the other hand, is novel in its use of fan-generated content to achieve near-prefect accuracies when combined with the audio techniques presented here and in [1, 2].

3. USE CASE

For the Joke-o-mat application, we assume the following use case: The first time a person watches a sitcom, the viewer needs hardly any navigation. Unlike other media, such as recorded meetings, sitcoms are designed for entertainment and should hold the viewer's attention for the entire length of an episode. Play and pause buttons should be sufficient. An involuntary interruption of the flow of the episode might detract from the experience. When a sitcom is watched at later times, however, a user might want to show a very funny scene to a friend, point out and post the sharpest punchline to Facebook, or even create a home-made YouTube video composed of the most hilarious moments of his or her favorite actor. In order to do this quickly, a navigation interface should support random seek into a video. Although this feature alone makes search for a particular moment in the episode possible, it remains cumbersome, especially because most sitcoms don't follow a single thread of narration. Therefore, the user should be presented with the basic narrative elements of a sitcom such as the scenes, punchlines, and individual dialog segments on top of a standard video player interface. A per-actor filter helps to search only for elements that contain a certain protagonist. Second-time viewers might, for example, like to search for all punchlines containing the word "soup" or all scenes about the "armoire".

4. INITIAL APPROACH

In the following, we briefly present more details on our initial approach as was presented in [2] since many of the elements of the initial system are re-used in the new system.

The initial system consists of two main elements: First, a preprocessing and analysis step; and second, the online video browser. The preprocessing step consists of an acoustic event detection and speaker identification step, the goal of which is to segment the audio track into regions, and a narrative element segmenting step. For the first step we distinguish the following types of events: Each of the main actors, male supporting actor, female supporting actor, laughter, music, and non-speech (e.g. other acoustic content). The speaker Gaussian mixture models are trained with both pure speech and laughter and music-overlapping speech. For the narrative element segmenting step we transform the segmentation into segments that reflect the themes, and generate icons for use in the graphical interface.

```
JERRY: I don't know. Uh, it must be love.
At Monks
========
PATRICE: What did I do?
GEORGE: Nothing. It's not you. It's me. I have a
  fear of commitment. I don't know how to love.
PATRICE: You hate my earrings, don't you?
```

Figure 2: Example of a Fan-sourced Script for *Seinfeld*.

While this initial approach is able to generate a decent narrative-theme navigation, it has certain limitations that led to the extension of the system as presented in the remainder of this article. First, the approach requires manual training of acoustic model for all the actors, who can vary episode by episode. It requires 60 seconds of training data per speaker, which can can be difficult to obtain for the minor roles. Most importantly, the approach does not take into account what was said, so it does not allow word-based operations such as search. Adding automatic speech recognition would be a possibility; practically this requires training of an acoustic model and a speech model, while our goal is to reduce the amount of training. Instead, we present a method that extends the approach while reducing the amount of required training and providing key-word-level search with an accuracy comparable to expert-generated annotation.

5. CONTEXT-AUGMENTED NARRATIVE-THEME NAVIGATION

The cost of producing transcripts can be quite high[1]. Fortunately, the huge growth of the Internet provides us with a new source of data in the form of scripts and closed captions produced by fans (generally the actual scripts are not available). However, this content is not directly usable for navigation. In this section, we describe a method of processing this "fan-sourced" data to produce an accurate transcript. The concrete realization described here is specific to a particular show (*Seinfeld*) and to the data we found online. However, the tools and methods presented should generalize to a wide variety of other content and tasks.

5.1 Fan-generated Data

Many shows have extensive fan communities with members who manually transcribe the episodes by listening carefully to the actual show as aired. Many of these fan-sourced scripts are available on the web, and most contain speaker-attributed text. See Figure 2 for an example.

For this work, we used the first hits from a Google search; we did not select for accuracy. However, we listened to ex-

[1]In previous work on transcribing multiparty meetings, we found that a one hour meeting could take upwards of 20 hours for a human to transcribe and there is no reason to think that a sitcom would be qualitatively different.

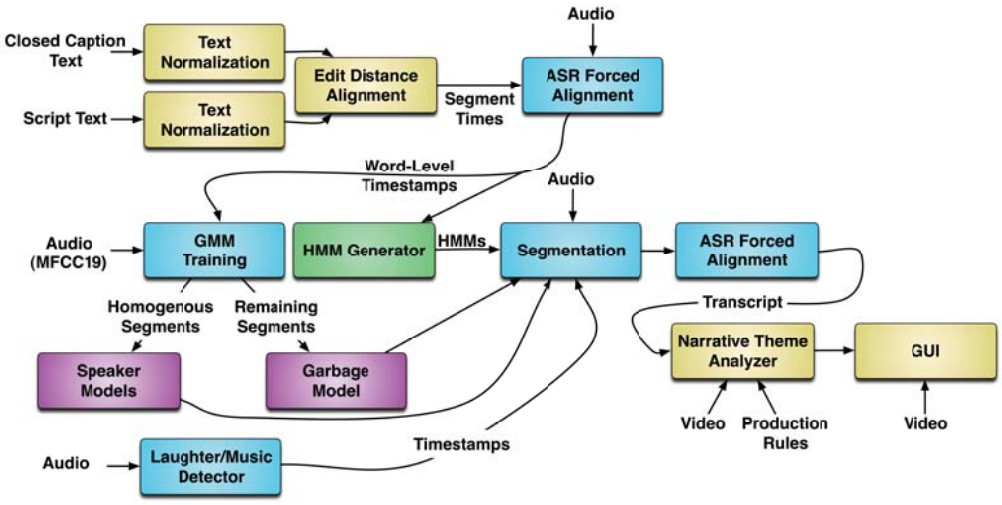

Figure 1: Overview diagram of the script- and closed-caption augmentation algorithm for narrative-theme navigation in sitcoms as described in Section 5.

```
00:04:52,691 --> 00:04:54,716
I don't know. It must be love.

00:05:04,136 --> 00:05:06,468
-What did I do?
-Nothing. It isn't you.
```

Figure 3: Example of Fan-sourced Closed Captions for *Seinfeld*.

cerpts, and found these scripts to be very accurate. They even include disfluencies (e.g. uh, um), which are useful for accurate speech recognition. However, there is no indication of when the particular words occurred within the episode, so we cannot cue the video to a particular location based on keywords with this data alone. Fortunately, there is another source of data. In order to accommodate deaf viewers, many programs contain closed captions. These do not typically contain speaker attribution–just the words being spoken. With the context of the images, this allows viewers to infer who is speaking. The closed captions seem to be less accurate than the fan-generated scripts, both from outright errors and because they are often intentionally altered (e.g. shortened to in order to be read along with the episode). As will be explained in Section 5.2, for speech recognition to detect the start times and end times of the words themselves, one needs what was actually said.

Neither the script nor the closed captions alone provide all the information needed. The scripts lack time information, and the closed captions lack speaker attribution. In the next sections, we describe a process that merges the scripts and the closed captions, and then uses speech and speaker recognition technology to determine the start time and end time of (almost) every word in the episode. This allows navigation of scenes by which actor is speaking and by keyword.

The first step is to normalize the fan-sourced data into a uniform format (e.g. spelling, hyphenation, punctuation, removal of scene descriptions and other non-audio text). This is very similar to the text normalization done for other corpora. Much of it was done automatically, although some cases would have required complex AI, so were hand corrected.

The closed captions we found on the Internet all appear to be generated using the OCR program *SubRip*, though with differing training data and setups leading to diffing output. The program introduced some interesting OCR errors, e.g. lower case "L" being used where capital "I" was intended. Fortunately, this is easy to correct automatically. Other errors, notably in numbers (e.g. "$ 1 9.45" when "nineteen dollars and forty five cents" was said) were relatively few, and were hand corrected.

Apart from the manual interventions described here and the initial training of music and laughter models (see Section 5.3), the system is automatic. The time to process a script and a closed captioning for an episode varies, depending mostly on the non-dialog structures (e.g. how scene transitions were marked). With a little bit of experience, an annotator can prepare an episode in about half an hour followed by about fifteen minutes of computer time.

5.2 Automatic Alignment

To understand what follows, we must first introduce the concept of *forced alignment*. This is a method from the speech recognition community that takes audio and a transcript, and generates detailed timing information for each word. Since forced alignments consider only one possible sequence of words (the one provided by the transcript), it is generally quite accurate compared to "open" speech recognition. For a given piece of audio, all steps of a speech recognizer are performed. However, instead of using a language model to determine the highest probable path, the results are restricted to match the given word sequence from the transcript. For the forced alignment to be successful and accurate requires several factors. As with general speech recognition, the closer the data is to the data on which the system was trained the better. For our system, we used SRI's *Decipher*, trained primarily on multiparty meetings

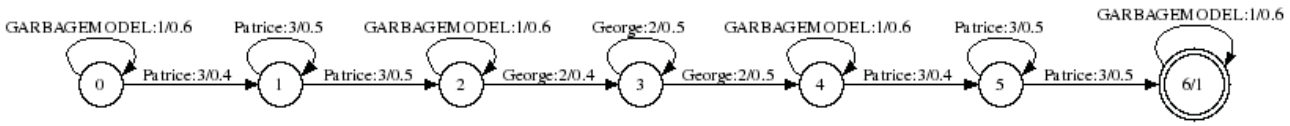

Figure 4: An example of an automatically generated Hidden Markov Model for resolving the alignment of a Multi-Speaker Segment. The model is inferred based on the fused fan-generated transcript and closed caption and used together with speech recognition and speaker identification to generate time- and speaker-aligned dialog segments. For further details see Section 5.2.

using head-mounted microphones. Clearly, there is a mismatch. An advantage of the recognizer is that it also does speaker adaptation over segments, so results will be more accurate if the segment contains only one speaker. Shorter segments typically work better than longer segments. Finally, forced alignment requires an accurate transcript of what was *actually* said in the audio. If a word occurs in the transcript that doesn't occur in the recognizer's dictionary, the system will fall back to a phoneme based model, and attempt to match the rest of the segment normally. Although it can handle laughter, silence, and other background sounds, it performs significantly better if these are removed.

Figure 1 shows the processing chain from fan-sourced data to finished GUI. The script and closed captions are first normalized. We then perform an optimal text alignment of the words in the two data sources using the minimal edit distance. For the scripts and closed captions we found on the web, this alignment frequently yields a segment where the start and end words from one line in the script match the start and end words from an utterance of the closed captions. In these cases, we use the start time and the end time of the closed caption, and the (single) speaker label and words from the script. In Figure 2 and Figure 3, Jerry's line is an example. We call this a *single-speaker segment*.

Many segments, however, are not so simple. Consider Figure 2 and Figure 3 again. The closed caption gives us the start time and end time of segments, but only for Jerry's line do we have a single speaker. For the remainder, the best we can say is that it starts at `00:05:04.136`, ends at `00:05:13.677` and is Patrice followed by George followed by Patrice. We do not know the internal boundaries of when Patrice stops talking and when George begins. We call this a *multi-speaker segment*. Of the ten episodes we processed, 37.3% of the segments were multi-speaker.

We run the forced alignment algorithm on all the segments. As described above, this can fail for a variety of reasons. For the 10 episodes we tested, approximately 90% of the segments aligned in the first step. For these segments, we now have for each word a start time, an end time, and a speaker attribution. For each actor, we pool the audio corresponding to all the words from the successful forced alignments and train a speaker model. We also train a similar *garbage model* on the audio that falls between the segments — we assume these areas contain only laughter, music, silence, and other non-speech audio content.

For the few failed single-speaker segments, we still use the segment start time, end time, and speaker for dialog-level segmentation, but lack a way to index the exact temporal location of each word (see Section 7 for a further discussion).

For each failed multi-speaker segment, we generate a Hidden Markov Model (HMM) that represents the particular order of speakers in the multi-speaker segment interspersed with the garbage model. For the example given in Figure 2 and Figure 3, the model allows zero or more frames of garbage, followed by some frames of Patrice, then zero or more garbage, then some George, then garbage, Patrice, and finally garbage. This is shown graphically in Figure 4. The initial state is state 0 at the start of the utterance. One then advances by moving across an arc, consuming one time step, and collecting up the probability given the model. For example, when the algorithm traverses from state 1 to state 2 across the "Patrice" arc, the speaker model for Patrice is invoked at that time step. Interspersing the garbage model allows us to account for segments that span non-dialog events, such as laughter, scene transition music, the door buzzer, etc.

An optimal segmentation is computed by conceptually traversing all possible combinations of paths through the HMM and outputting the most probable path.

One potential problem with the segmentation method described so far is that it depends on the garbage model not containing any audio from actual speakers. If it accidentally does contain audio from speakers, then the garbage model could match audio that should be attributed to an actor. To mitigate this, we impose a minimum duration for each speaker. For example, the HMM in Figure 4 has a minimum duration for each speaker of three frames (e.g. there is no way to go from state 2 to state 4 without consuming three frames of George's speech). Informal experiments on one episode showed very little sensitivity to minimum duration, indicating that the garbage models likely contain little speech audio. For the actual algorithm, the minimum duration was set to 50 frames (0.5 seconds).

At this stage of the algorithm, we have the start time and end time for each speaker (and garbage sections) for the failed multi-speaker segments. In essence, we have converted a multi-speaker segment to several single-speaker segments. Since these new segments contain less garbage and are single speaker, the forced alignment step should perform better on them. We therefore run forced alignment again and process the results the same way as for other single-speaker segments.

5.3 Music and Laughter Segmentation

The fan-sourced scripts and closed captions do not contain any systematized information about where laughter and scene transitions occur. However, this is a vital component for browsing sitcoms. For detecting laughter, we use the open-source speech decoder *Shout* [9] in speech/nonspeech mode. Since dead air is anathema to broadcasters, almost

False Alarms	14.0 %
Missed Speech	8.2 %
Speaker Error	2.4 %

Table 1: Diarization Error Rate between fan-sourced and expert-generated annotations as explained in Section 6. Word-alignment accuracy was not measured.

Prefer Fan-augmented	16 %
Prefer Expert-generated	12 %
No Preference	72 %

Table 2: User preferences for the automatically generated transcripts augmented with fan-sourced scripts and expert-generated annotations. For more details see Section 6.

everything Shout detected as non-speech is laughter. Fortunately, the few segments that Shout incorrectly marked as laughter were almost exclusively quite short (often a few notes of music, the door buzzer, etc.), and are therefore not used in the interface. Since our primary interest in laughter is to use its duration as an indication of how funny a punchline is, we actually don't even use segments marked as laughter that are short in duration. For music detection, we used pre-trained models as described briefly in Section 4. An obvious extension would be to use visual cues in addition to music detection for scene transitions.

5.4 Putting It All Together

The combination of normalization, text alignment, forced alignment, HMM segmentation, and laughter detection yields the start time and end time of each speaker in the script and the start time and end time of almost all the words in the script (minus the words of single-speaker segments that failed to align). The events are used as input to the Narrative Theme Analyzer. Figure 5 presents the final version of the navigation as shown to the user. Together, this allows us to use only the video and data found on the web plus a small amount of time spent normalizing the data and training a laughter and music detector. Although the realization presented is specific to the particular "found" data, the techniques described are applicable to a wide range of tasks where incomplete and semi-contradictory data are available.

6. EVALUATION

Anecdotally, the generated alignment based on the algorithm presented here is very close to ground truth. In fact, ground truth for many corpora are generated in a similar way, although instead of fans, experts are used for the annotation. Therefore, one way to evaluate the quality of the fan-sourced data is to compare it to "expert" annotation. We measured this inter-annotator agreement in two ways: The time-based Diarization Error Rate (DER) and a human subject study.

DER for the *Seinfeld* episode *The Soup Nazi* for the fan-sourced annotations scored against the expert-generated annotations is presented in Table 1. The false alarms appear to be caused by the fact that the closed captions often span several dialog elements, even if there is some amount of non-dialog audio between the two dialog elements. As a result, the fan-sourced annotations include some non-speech in the middle of single-speaker segments that the expert marked as two distinct dialog elements. Many of these small non-speech pieces add up to a fairly significant number. The comparison of two human annotators might have resulted in the same error. For that reason, and because of the lack of expert annotation of the words, we refrained from measuring word-alignment accuracy. The effect could be reduced by simply running a speech/nonspeech detector on the final single-speaker segments and excluding the nonspeech regions. However, as will be shown in the next Section, it is unclear if this is necessary. Missed speech appears to be an artifact of the forced alignment process, which sometimes truncates words at the end of an utterance more abruptly than an expert would do. This could be reduced by padding the end of each utterance by a small amount, possibly at the expense of increasing false alarms. The missed speech may also be caused by backchannels ("uh huh", "yeah") that the expert marked, but the fan-sourced scripts did not include. The very low speaker error rate indicates that when both annotation methods indicate that an actor is speaking, music is playing, or the (canned) audience is laughing, they agree.

6.1 User Study

To measure whether the differences between the fan-sourced and expert-generated annotations are relevant to the Joke-o-mat application, we performed a user study with 25 participants. A web site presented the user with two versions of the Joke-o-mat interface, identical except that one was generated from the expert annotations and the other from the fan-sourced annotations. The order in which the two versions were presented was randomized for each visitor. The subjects were asked to browse the episodes, and then select which version they preferred or "no preference". The results are shown in Table 2.

Most users expressed no preference between the fan-sourced and expert-generated annotations. Those that did express a preference were almost evenly split between the two. We conclude that, at least for the Joke-o-mat application, there is no substantive difference between the two methods of generating the annotations.

7. LIMITS OF THE APPROACH AND FUTURE WORK

Several of the limitations of the initial approach have been addressed here, especially the previous need for extensive manual training of speaker identification and speech recognition. The new method does not need any further manual labeling of actor names. However, the problem of insufficient training data for certain supporting actors still remains. Also, laughter and scene transition music still mostly have to be trained manually. Obviously, the method fails when there are no closed captions and/or scripts. However, for commercial use, the original scripts should be acquirable from the original authors and closed captions are virtually always available, at least in the United States and Europe. Future work will include genres other than sitcom, since many follow similarly strict patterns of narrative themes and also have fan-provided content on the Internet (e.g. dramas

and soap operas). We would expect to edit the rules of the narrative theme analyzer for other genres. In the long run, it would also be interesting to generalize the idea to non-broadcast media, such as lectures and seminars where (several) imperfect transcripts should be available. Finally, computer vision techniques would add many more possibilities and also improve the accuracy of scene detection.

A technical problem still remains with how to handle failed single-speaker segments. When a single-speaker segment fails to align, we currently still use its start time, end time, and speaker label for dialog segmentation. We do not use the words, since the interface assumes we know the start time and end time of each word. An easy extension would be to simply interpolate the time to approximate the location of the word. For example, a word that appears halfway through the text of the segment could be associated with the time halfway through. This would almost certainly be close enough for keyword spotting. Another approach to single-speaker segments that fail to align is to retrain all the speaker and garbage models using the final segmentation and iterate the whole process. Since the final segmentation includes more data, the models should be better. An exciting and more sophisticated approach would be to train an HMM similar to Figure 4, but over the entire episode, and with a laughter model separate from the garbage model. The speaker models and laughter models could be trained on segments derived from the processes described in this paper. An optimal path through the full-episode HMM would identify not only where each speaker started and stopped speaking, but also the laughter, and would not depend on the accuracy of the script or the closed captions, only on the accuracy of the models. Of course, errors would be introduced because machine learning is not perfect. It is an open question what is the trade off between tolerating errors in the fan-sourced data vs. introducing errors through machine learning. Exploring this trade-off would be especially interesting for more errorful data.

8. CONCLUSION

This article presented a system that enables enhanced navigation of sitcoms episodes. Users can navigate directly to a punchline, a top-5 punchline, a scene, or a dialog element, and can explicitly include or exclude actors in the navigation and search by keyword. The method for producing the segmentation leverages the artistic production rules of the genre, which specify how narrative themes should be presented to the audience. The article further presents an extension that generates word-level transcripts by augmenting the speaker identification step and a speech recognition step with a combination of fan-generated scripts and closed captioning. An evaluation of the approach shows the system to be performing with nearly ground-truth accuracy.

Acknowledgements

This research is partly supported by Microsoft (Award # 024263) and Intel (Award # 024894) funding and by matching funding by U.C. Discovery (Award # DIG07-10227).

9. REFERENCES

[1] G. Friedland, L. Gottlieb, and A. Janin. Joke-o-mat: Browsing sticoms punchline-by-punchline. In *Proceedings of ACM Multimedia*, pages pp. 1115–1116. ACM, October 2009.

[2] G. Friedland, L. Gottlieb, and A. Janin. Using artistic markers and speaker identification for narrative-theme navigation of seinfeld episodes. In *Proceedings of the 11th IEEE International Symposium on Multimedia*, pages pp. 511–516, December 2009.

[3] S. fu Chang, W. Chen, H. J. Meng, H. Sundaram, and D. Zhong. A fully automated content-based video search engine supporting spatiotemporal queries. *IEEE Transactions on Circuits and Systems for Video Technology*, 8:602–615, 1998.

[4] M. Larson, E. Newman, and G. Jones. Overview of videoclef 2008: Automatic generation of topic-based feeds for dual language audio-visual content. In *Working Notes for the CLEF 2008 Workshop*, Aarhus, September 2008.

[5] NIST Rich Transcription evaluation. http://www.itl.nist.gov/iad/mig/tests/rt/.

[6] NIST TRECVid evaluation. http://www-nlpir.nist.gov/projects/trecvid/.

[7] C. G. M. Snoek and M. Worring. Concept-based video retrieval. *Foundamental Trends in Information Retrieval*, 2(4):215–322, 2009.

[8] H. Wactlar, T. Kanade, M. Smith, and S. Stevens. Intelligent access to digital video: Informedia project. *Computer*, 29(5):46–52, 1996.

[9] C. Wooters and M. Huijbregts. The ICSI RT07s Speaker Diarization System. In *Multimodal Technologies for Perception of Humans: International Evaluation Worksho ps CLEAR 2007 and RT 2007, Baltimore, MD, USA, May 8-11, 2007, Revised Selected Papers*, pages 509–519, Berlin, Heidelberg, 2008. Springer-Verlag.

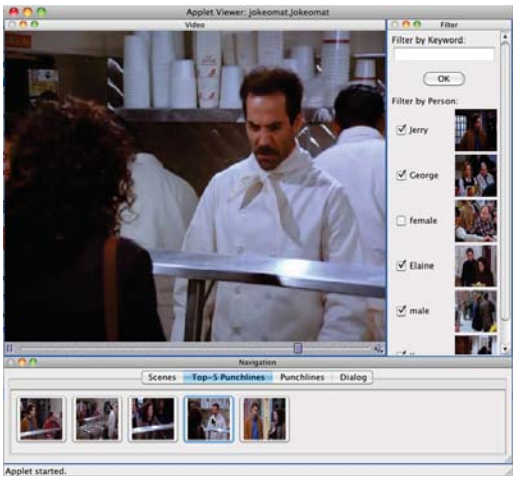

Figure 5: Our sitcom navigation interface: Users can browse an episode by scene, punchline, and dialog. The top-5 punchlines are shown in a separate panel. Actors can be selected and unselected and keywords can be entered that filters the segments shown in the navigation. All navigation elements are extracted automatically by aligning speaker identification and speech recognition output with closed caption and fan-generated scripts.