9-6-2013

Data Collection for the Similar Segments in Social Speech Task

Nigel G. Ward
The The University of Texas at El Paso, nigelward@acm.org

Steven D. Werner
The University of Texas at El Paso, stevenwerner@acm.org

Comments:

Technical Report: UTEP-CS-13-58

UTEP Technical Report UTEP-CS-13-58

Data Collection for the Similar Segments in Social Speech Task

Nigel G. Ward and Steven D. Werner

Department of Computer Science
University of Texas at El Paso
500 West University Avenue
El Paso, TX 79968-0518

email: nigelward@acm.org, stevenwerner@acm.org

September 6, 2013

Information retrieval systems rely heavily on models of similarity, but for spoken dialog such models currently use mostly standard textual-content similarity. As part of the MediaEval Benchmarking Initiative, we have created a new corpus to support development of similarity models for spoken dialog. This corpus includes 26 casual dialogs among members of two semi-cohesive groups, totaling about 5 hours, with 1889 labeled regions associated into 227 sets which annotators judged to be similar enough to share a tag. This technical report brings together information about this corpus and its intended uses, previously only available on the project website.

Index Terms: information retrieval, multimedia, corpus, video, audio, speech, dialog, search, recommendation systems, MediaEval

1 Goals

Information retrieval systems, recommendation systems, and other language processing systems rely heavily on models of similarity, both document-document and document-query similarity. The development and evaluation of these similarity models requires suitable training corpora. For spoken dialog, such resources exist only for a few genres [Ward and Werner, 2012]. Moreover, the design of these corpora has been based on certain assumptions, for example, that the input is monolog, that it is deliberate speech, that it is clearly divided into topics, and that it can adequately be indexed using the terms alone [Ward and Werner, 2012, Ward and Werner, 2013]. Therefore we decided to create a new corpus, with the aim of support the development of systems that overcome these assumptions.

Out of the many possible genres, we chose to collect informal dialogs between members of the same loose social group. We call this "social speech" as it is the spoken-dialog analog of the

sorts of things found in social media. This genre was chosen not only because of the potential value [Ward and Werner, 2013, Ward et al., 2013] and the timeliness of the topic, but also to be maximally non-redundant to existing corpora. Social speech also serves as a useful proxy for many other dialog genres [Ward and Werner, 2012].

While the corpus is interesting, the real value and novelty here is in the annotations. These provide indications of which sets of dialog regions are similar.

Wanting from the start to share this data widely, we fortunately connected with the organizers of MediaEval, the Benchmarking Initiative for Multimedia Information Retrieval. After a formal proposal, MediaEval accepted search in this corpus as one of the 2013 challenge tasks. For the task we defined a novel evaluation method, incorporating a simulation of user behavior in a query-by-example scenario. We also created transcripts and prosodic features sets, to lower the barrier to entry for teams considering participating in the challenge.

The potential uses of this corpus extend beyond MediaEval. It can also support other evaluation methods and other use scenarios. In particular, it was also designed to support user studies. As the topics in this corpus include many of interest to students, it is easy for us, and for a lesser extent to other researchers with access to populations of college students, to measure, with real, motivated subjects, the actual value to users of complete systems for search or content recommendation.

This technical report brings together information about this corpus and its intended uses. It is intended for those who have already read the task overview paper [Ward et al., 2013] and want more detail.

2 Scenario and Task

While we later came to focus on support for building similarity models, originally our thoughts were focused on the construction of complete systems able to actually support users in a realistic scenario. This scenario and the task are as described in Sections 1 and 2 of [Ward et al., 2013].

3 Data Collection

The data collection is summarized in Section 3 of [Ward et al., 2013]. This section provides more detail.

3.1 Recording Conditions and Equipment

Dialogs were recorded with the conversants in different rooms seeing each other through a glass wall.

Wanting stereo with good acoustic separation between the rooms, hence the need for a wall, but also wanting to gather video, we recorded in a pair of rooms with a connecting glass wall. The resulting acoustic separation was not perfect, but the small amount of across-track bleeding is not noticeable at most listening volumes. The video was also not perfect, with some reflections and awkward angles, but acceptable in quality, at least to human eyes.

Each participant wore a Sony DR-200 headset. The inputs were fed into an Olympus DS-2 digital voice recorder. Audio from monitoring jack of the recorder was split, to 1) sent it back to the participants' headsets so they could hear each other 2) feed the audio input of our main recording device, an iMac. The recordings were created with Quicktime on the iMac, with the video coming from the built-in camera and the audio from the voice recorder. Subsequently the recordings were edited with iMovie to remove the times where subjects were being set up and when the recording was being ended.

The first recordings were done a Macbook Pro, which had a better camera. However it lacked an audio input jack, and attempts to synchronize in the separately recorded audio using iMovie were unsatisfactory because of the inability to get exact alignments and because of clock skew. Next we used Photobooth, but discovered that it would abort the recording if a system alert popped up. We therefore settled on Quicktime, despite the fact that its narrow aspect ratio wasn't ideal for our through-the-window setup.

3.2 Notes on Specific Dialogs

Those wishing to get an intuitive feel for the corpus are recommended to start by listening to dialogs 004, 006, 008, and 012, as these are rich in interesting and diverse audio regions of the sort that one might want find as results to a search.

The release includes all English-language dialogs collected, including some with flaws and some not strictly comparable to the others:

- Dialogs 000–002: recorded using a MacBook (whereas the body of the corpus was recording using an iMac).
- Dialogs 000,002: include participants knowledgeable about our systems, processes, and information retrieval technology.
- Dialogs 014, 016–018: include one non-CS major (although he/she was taking a CS class).
- Dialog 000–002, 004, 007: audio recorded separately from the video and later imperfectly aligned.
- Dialogs 000, 004, 005, 010, 011, 013–015: video incomplete.
- Dialogs 001 and 007: topic suggestions were given strongly and/or participants took them literally.

One of the dialog participants offered to speak Spanish, a few dropped in a few Spanish words, and several were not native English speakers, as indicated in the metadata.

3.3 Other Information

While the dialogs were solicited, all of them were among people who might have had a conversation anyway that day. Most of the speakers were engaged in their conversations, most had to be stopped when the time was up, and several remarked that they gladly would have continued talking. Overall the dialogs were quite natural.

Appendix A, the official Data Collection Protocol, as submitted to the UTEP Institutional Review Board (human subjects experimentation committee), overviews the collection.

Appendix B is a flier used to recruit subjects.

Appendix C is the explanation that the subjects read and the consent form they signed. Notable are the restrictions on the use of the data, which must be respected.

Appendix D is the verbal instructions given to the subjects.

Appendix E is the metadata for the dialogs, including for each dialog information on the participants including gender, age range, class status, native languages, and prior relationship to the interlocutor.

4 Annotation

The annotation is summarized in Section 3 of [Ward et al., 2013]. This section provides more detail on who did the tagging and on the different tagging styles observed.

4.1 Annotators

For the training set:

- Annotator 1 was speaker 6. Atypically he did the annotations before the Annotators Guide was available, he's not a student, and he's not naive about information retrieval technology.
- Annotator 2 was speaker 1. Atypically he knows a lot about one technique of possible value for this task.
- Annotator 3 was speaker 28. Atypically she's not quite part of the CS community, being a math major, although she was taking a CS course and does IT work in her job.
- Annotator 4 was speaker 2. Atypically, she has experience doing search in audio archives, having spent about 10 hours doing so as part of the experiment reported in [Ward and Werner, 2013].
- Annotator 5 was speaker 30.
- Annotator 6 was speaker 5. Atypically, he was a member of the research lab, having joined a few months ago, and somewhat knowledgeable about our favorite analysis methods.

For the testset:

- Annotator 1 was the same as annotator 3 for the testset.
- Annotators 2 and 3 were CS undergraduates. They worked at the the same table, and may have shared some thoughts on the tags.
- Annotator 4 was a high-school student.

4.2 Tagging Styles

The most salient differences in tagging styles involved the length of the tagged regions.

Compared to the trainingset tags, there were many fewer long regions tagged in the testset. In the training set there was one region tagged over 4 minutes long (tv-shows), four more over 3 minutes (playing-video-games, programming-projects, movies-tv-shows, course-experiences), and 58 over two minutes. In the testset there were only 5 longer than 2 minutes, and these were all for just one category of one annotator. One reason for the difference is that, having noticed the over-long tags in the training set, we suggested to the testset annotators that it was okay to leave large sections of the data without any tags, and that could be appropriate to break up a long region on a single vague topic (for example entertainment) into more specific contributions. Another possible factor is that, with only 6 conversations to work on, the testset annotators may not have felt the same degree of desire to get it over with.

There was a small problem with short tags: most of the regions tagged by testset annotators 2 and 3 turned out to be only 1-4 seconds in length; it seemed that they had interpreted their job as being the identification of semantically related words in the corpus. While this relates to a potentially interesting task, it is different from the one in our scenario, and so their tagsets were dropped.

4.3 Other Information

The relevant appendices are:

Appendix F, the Annotator Training Overview, lists the steps used to train the first six annotators.

Appendix G, the Annotators Guide, specifies the goal, guidelines, and procedure for annotation.

Appendix H, How to Tag Social Speech using Elan, explains the software used.

5 Features

In order to help teams participating in this task, we provided both transcripts and prosodic features.

5.1 Transcriptions

Initial attempts to obtain transcripts using a free Google transcription service and using Dragon Dictate gave outputs that were correct for fewer than one word in ten.

The actual transcripts were kindly provided by Steve Renals of the University of Edinburgh. According to his note:

> These were produced by a system which should be considered to be very much a baseline system using acoustic models trained on meetings data (primarily AMI cor-

> pus) and a language model optimised for NIST RT meeting transcription. For each recording there's a "rec" file, giving the detected speaker, start and end times (in centiseconds) and transcript for each utterance, and an "mlf" file, with detailed timings of words and phones within the utterances (but the utterances not in chronological order). We imagine it will be mainly the "rec" files that participants want to use.

In addition we provided human-generated transcripts. The bulk were done by trainingset annotator 3, with the 20, 27, and the first half of 26, done by an inexperienced volunteer. They used TranscriberAG. The instructions were:

> What we want is each of the 26 dialogs transcribed (all dialogs except the Spanish ones.) We're using Transcriber because it gives timestamps at the utterance level; so please select roughly utterance-sized regions (probably 4-10 seconds) and label each one.
>
> Speed is more important for this task than accuracy. In particular, it's okay to skip nonlexical items (uh-huh, um, and laughter), word fragments, and anything that's not clear on first listening. (Although it's also okay to include them, at times when that's more convenient.) If there are technical terms or other words you can't catch, just leave them out or enter xxx as a placeholder. Spelling errors should be avoided, but other minor errors are fine; you should never go back and correct mistakes.
>
> It's okay to be sloppy here because the aim is to be as good as powerful future automatic speech recognition, but not perfect, since that would be unrealistic for us when we test the systems.

5.2 Prosodic Features

There are three sets of prosodic features.

5.2.1 Basic F_0 (Pitch) Values

These were generated using the Hirose-Seto algorithm [Hirose et al., 1992]. There is a f0 file for each track, left and right, of each conversation. Each file has one line every 10 milliseconds. Each line in turn has three fields: a timestamp in 100-nanoseconds, the most likely pitch value, and an boolean flag indicating whether or not pitch was in fact likely to be present at that point.

5.2.2 78 Contextual Prosodic Features

Out of the thousands of possible prosodic features that one might consider, this is a set of the sort that we have found helpful for characterizing dialog activity. They are track-normalized, computed over fixed windows (rather than being utterance-, word-, or syllable-aligned), and computed at various offsets. More about such features, including reasons for using them, how they are computed, and experiences with them are described elsewhere [Ward and Vega, 2012, Ward and Werner, 2013].

Specifically for each conversation in the corpus there are two .pc (prosodic context) files, one for the right track and one for the left. Both contain one line for every 10-milliseconds of the conversation. Each line specifies the values for 78 features. Each line in that file below indicates the base feature type, then the start time of the window over which that feature is being computed in milliseconds relative to the current frame, the word "to", the end time, and the track. The abbreviations are:

vo volume

ph pitch height

pr pitch range

sr speaking-rate proxy

self speaker in this track

inte speaker in the other track (interlocutor)

Appendix I, slimcrunch.fss, lists the items in this featureset.

5.2.3 78 prosodic dimensions

These are PCA-rotated composite features, derived from the above set of 78 original features.

6 Evaluation

The evaluation method is summarized in Section 3 of [Ward et al., 2013].

One complexity is that our idea of evaluation by user simulation is not supported perfectly by our tagset-based algorithm, since region-pairs which do not share a tagset may still be similar. While the metrics are adjusted for this, this remains the biggest issue with our evaluation method.

Appendix J, Guide to Interpreting the Metrics, discusses the realism and stability of the main measures, explains the normalization factors, and describes the ancillary measures.

Appendix K, Baselines on the Testset, gives the performance of a random baseline and a clever baseline.

Appendix L, score5.py, is the python code that computes the performance metrics.

7 Availability

The recordings and transcripts are available by request to either of the authors.

The tagsets, scoring software, prosodic features, and documentation are also available at http://www.cs.utep.edu/nigel/ssss/ .

All of the above are also archived in the UTEP Library Special Collections Section.

8 Frequently Asked Questions

This section addresses some questions asked by task participant teams.

How does this differ from topic detection?

Some of the similarity sets will probably be largely or entirely topic-based, but most will probably also involve on other factors, such as user goals (talk about childcare experiences in order to find a new daycare provider, versus in order to find ways to make the child feel comfortable), and such as attitudes and purposes (talk about a course for the sake of deciding whether to take it, or for figuring out how best to study for it, or for just sharing stories about the professor).

What exactly do systems have to do?

Given a query (in the form of a region of one of the dialogs), the system should return a set of pointers to hypothesized similar regions. The ideal system will return the onset points of all such similar regions, as identified by the human annotators, and no other regions.

Could you provide us some more details on how a baseline system would work?

One the obvious way to build a baseline system would be to gather all the words in the query region, then find all regions elsewhere in the corpus which densely contain those words or similar words. For this any traditional IR technique would work, although probably with modifications to deal with the special properties of spoken language (noisy, interlocutor-track information available, prosodic features also available), and with the lack of a segmentation into "documents", meaning that system can return regions from anywhere in the corpus and of any size.

Will the final dataset contain a fixed number of tags?

No. In fact, the tags are there as comments only. The system you build will not be able to rely on the tags being there or meaning anything. The meaning of each similarity-set is just the set of regions in that set. And in particular, the test set will include as queries regions which were not seen in the training set, and which may not relate to any of the tags seen in the training set. We think this is realistic. For example, our campus recently had a bomb threat, the first in 10 years, so there is no talk anywhere in the corpus about anything like that, but we'd still want a system running today to be able to find other regions of talk about campus security issues if a segment related to a bomb threat was submitted as a query.

How many training segments will there be available for each tag? Right now, we are seeing tags that only have 1-2 training segments; we do not see how we can train classification models from these segments.

Training a classifier for each tag would be a poor strategy, since the tagset is not fixed. The goal of the task is to find similar segments, and the similarity-sets provided as examples of what counts as similar in this corpus, for these users. If you use these to build and refine a general similarity metric, then that similarity metric can be used for any retrieval request. For example, if you use a vector-space model, then for any query (e.g. a couple of utterances about the bomb threat), you can find some speech regions that are close to it in the vector space, and return those. In a real system you'd probably use a nearest-neighbors algorithm to find these quickly, but for this task, due to the small corpus and lack of a real-time requirement, exhaustive search

will probably be just fine. (But you're right to note that having only 1 example of some tags is not useful for anything; in the actual data release we'll aim to have 5-15 examples for most tags.)

Will you provide a training, development, and test set? Right now, there is only mention of a training and testing set.

While most teams will want to split the training set themselves into one part for training and one part for tuning, we aren't imposing any specific partition.

Why did the developers themselves contribute to the recordings and annotations? Mightn't that skew things somehow?

The test data will be pristine, so there is no risk. However for training purposes we decided to release also the pilot recordings (000–002)and annotations, thinking that participants would like to have as much data as possible. However the metadata shows which files these are, so it's possible to exclude them from training if desired.

Why are the value weights for?

As described in section step 7 of the Annotators Guide, each similarity set over the training data was assigned a value, from 0 to 3. These numbers may be useful in the training process, since similarity sets ones with higher values may be more informative/valuable, and participants may want to tune their parameters to perform best on the higher-valued sets.

Some regions appear in multiple places in the tagsets; is this because some segments were assigned multiple tags?

Yes, and this is why the key performance metrics are adjusted, as described in the Guide to Interpreting the Metrics document.

There are some similarity sets that are semantically very much close to each other but are nevertheless seen as different similarity sets. For example, the sets #Courses, #courses, #course material, and #Course Work are all very similar but in evaluation (if you test with all possible queries over the training set), the system will be penalized if, for example, a test query from #Courses results in retrieval of a segment from #courses. One could consider merging some of these simsets in order to obtain a more accurate view of the system's current performance.

Yes, that's true. However we are reluctant to try merging the simsets, as that would involve a lot of subjectivity, and would lose some information.

If this is a problem in training, task participants may consider reducing the penalty for false alarms (falsePositivePenalty in score5.py), since, as noted, many of the false alarms are not really errors, at least during the early stages of training.

Acknowledgments

We thank our dialog participants, annotators, and transcribers, especially Jeanette M. Mendoza.

Martha Larson helped us formulate the problem as a challenge task.

Khiet Truong found several bugs in the initial releases, and posed most of the questions above.

Steve Renals provided the automatic-speech-recognition transcriptions..

The Similar Segments in Social Speech organizing committee, Elizabeth Shriberg, Catharine Oertel, Tatsuya Kawahara, David Novick, and Louis-Philippe Morency, provided encouragement and advice.

Shirley Moore, Olac Fuentes, and Kay Roy helped us recruit participants for the dialogs.

Alejandro Vega, Shreyas Karkhedkar, and David Novick provided infrastructure and support.

This work was supported in part by the NSF under project IIS-0914868 and REU supplements thereto.

References

[Hirose et al., 1992] Hirose, K., Fujisaki, H., and Seto, S. (1992). A scheme for pitch extraction of speech using autocorrelation function with frame length proportional to the time lag. In *1992 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages I–149–152.

[Ward and Vega, 2012] Ward, N. G. and Vega, A. (2012). A bottom-up exploration of the dimensions of dialog state in spoken interaction. In *13th Annual SIGdial Meeting on Discourse and Dialogue*.

[Ward and Werner, 2012] Ward, N. G. and Werner, S. D. (2012). Thirty-two sample audio search tasks. Technical Report UTEP-CS-12-39, University of Texas at El Paso, Department of Computer Science.

[Ward and Werner, 2013] Ward, N. G. and Werner, S. D. (2013). Using dialog-activity similarity for spoken information retrieval. In *Interspeech*.

[Ward et al., 2013] Ward, N. G., Werner, S. D., Novick, D. G., Kawahara, T., Shriberg, E. E., Morency, L.-P., and Oertel, C. (2013). The similar segments in social speech task. In *MediaEval Workshop*.

UTEP IRB Research Proposal

I. Title: Turning Multimedia into Social Media: Data Collection

II. Investigators (co-investigators)

Nigel Ward, Ph.D., Department of Computer Science (Steven D. Werner, CS)

III. Hypothesis, Research Questions, or Goals of the Project

The goal of this data collection is to create a moderate size set of recorded dialogs which will support our research in dialog behaviors over the long term. In the short term we have three main research questions:

a. How can we support search in multimedia, specifically multimedia shared within social groups?
b. What are the recurring patterns of interaction in dialog, and how do these differ between English and Spanish?
c. What patterns of interaction are present in dialog, how can we model them and how can we build spoken dialog systems have more human-like behavior?

IV. Background and Significance:

Please see the attached three NSF proposals and one conference paper.

V. Research Method, Design, and Proposed Statistical Analysis:

Once we collect the data, we will subject it to an eclectic battery of analysis methods, including automatic analysis of various phonetic and prosodic features, statistical analyses over those features, the construction of predictive algorithms and similarity metrics, and analysis of failures of our models. These are outlined in the NSF proposals, and also in the papers that can be found at http://www.cs.utep.edu/nigel/pubs.html, especially A Bottom-Up Exploration of the Dimensions of Dialog State in Spoken Interaction (2012), Temporal Distributional Analysis (2011), and A Case Study in the Identification of Prosodic Cues to Turn-Taking: Back-Channeling in Arabic (2006).

VI. Human Subject Interactions

A. We wish our dialogs to include a variety of participants, who are related in a variety of ways. The important dimensions re:

1. familiarity between the participants
2. status difference between the participants
3. language spoken by the participants

In addition we will record the gender and native languages of the participants, but will not do anything active to control for these.

As we are interested in social-media type applications, we will recruit all participants from one quasi-social group, the Computer Science academic community.

We expect the main recoding effort to start in late March and be completed by mid April, with possible follow ons.

B. Our main recruiting population will be the students of CS 1401. These students have a "research experience" requirement, which they can fulfill in one of several ways, including serving as participant in an experiment.

Our second recruiting population will be all other students in computer science, with a focus on upper division students. We may advertise, for example, in CS 3331, being taught now by the PI, or post fliers (attached) in the open laboratory and on bulletin boards.

C. Informed consent will be obtained, and documented using the attached form. Please note that we are not using the standard template, because we feel that it is important for the subjects to have a concise, readable description which focuses on what matters, namely the privacy issues.

D. The data collection protocol is simply (to quote from the flier)
It's easy (total estimated time involved 20 minutes)
1. Sign up for a slot
2. Come to the ISG lab at the assigned time (room 1.0404, to the left after you enter the main lobby)
3. Fill out the consent and demographic forms (attached)
4. Enter the recording studio and wear a head-mounted microphone
5. Talk about anything for 5-10 minutes while being audio/video recorded
6. Get debriefed and learn about our research if you like, and DONE

E. We will maintain privacy, to the extent possible compatible with the aims of the data collection, by following the following rules (quoting from the consent form):
"Users of the Data: The data collected will be used by researchers in UTEP's Interactive Systems group and by researchers at other institutions who sign an agreement to follow all the confidentiality and privacy protections described below.
Uses of the Data: 1) After we collect the dialogs, we will recruit CS students to tag them for topics and other aspects that people might like to search on. 2) The collection of dialogs, together with these tags, will be analyzed here at UTEP's Interactive Systems Group and by researchers at other institutions. In general this analysis will be based on statistical analysis of automatically extracted features, but the researchers may listen to or view some of the data. Based on these analyses, researchers will develop an evaluate algorithms and methods for searching in or otherwise processing social multimedia 3) Excerpts of the data may be used in presentations at research meetings to illustrate these algorithms and methods. 4) Transcripts of small dialog segments may be used in scientific articles describing the algorithms, methods, and findings. 5) If you permit by initialing below, short audio snippets of the data (no video), not to exceed 8 seconds, may be placed on websites as illustrations of common patterns of interaction and how they differ in English and Spanish, or otherwise made publically available for educational purposes.
Risks and Confidentiality: The risks involved in participating in this study relate to the loss of privacy and the potential for embarrassment, in particular, since the taggers will be your peers and maybe your friends. To mitigate these risks, your name will not be associated with the dialog(s) you participate in. Despite this, taggers and other local users of the data may still be able to identify you from the video or your voice. We therefore request that you *please avoid saying anything that might embarrass you or anyone else.* If you inadvertently say something that should not be retained, please immediately make a note, and tell the experimenter after the dialog ends, so that he can

delete those segments of the audio and video. The experimenter, at his discretion, may also delete short segments if he thinks there is a potential privacy risk. All recordings will be stored only on password-protected computers and all backup media in locking file cabinets."

F. Our confidentiality protections are described above in part E.

G. We have recording equipment, two rooms separated by a glass window, and a waiting room.

VII. The risks and the mitigations are discussed above under VI.E

VIII. The benefits to participants are class credit or cash, plus the chance to meet interesting new people and talk about their major. The potential benefits to society are enormous, as documented in the three NSF proposals.

IX. No other sites are involved in the data collection.

X. This project is not subject to review by any other IRB.

Appendix B: Flier

CS Students!

Research Experiment = research credit || $

Help our research! Come and have a conversation about your major.

The next big thing in social media may be social multimedia, that is, audio and/or video recordings of you and your friends, their friends, friends of friends, and so on. One big obstacle, however, is the difficulty of finding information in such recordings. We are developing new algorithms for searching conversation, but to test them we need dialog data, so come talk and let us record you.

It's easy (total estimated time involved 20 minutes)

1. Sign up for a slot
2. Come to the ISG lab at the assigned time (room 1.0404, to the left after you enter the main lobby)
3. Fill out the consent and demographic forms
4. Enter the recording studio and wear a head-mounted microphone
5. Talk about anything for 5-10 minutes while being audio/video recorded
6. Get debriefed and learn about our research if you like, and DONE

Compensation for Participation

Research-Participation Credit:
CS 1401 students:
You will receive one Research-participation credit for the CS 1401 course.

Flat Rate:

Graduate students:	$8
CS 4310 or 4311 students:	$6
Others:	$5

Contact:

Steven Werner
(315) 405-3311 (txt or call)
sdwerner@miners.utep.edu
Interactive Systems Group UTEP CCS 1.0404

Appendix C: Consent Form

University of Texas at El Paso

Informed Consent Form

Project Name: **Turning Multimedia into Social Media: Data Collection**

Investigators: Steven Werner, Nigel Ward Ph.D.

Research Aims: We are collecting dialogs to generally support our research in speech technology. We are specifically collecting among UTEP CS students for three reasons: 1) to enable the development of algorithms for search in "social" multimedia data, 2) to discover the most common patterns of interaction in casual conversation, and 3) to illustrate these patterns and their difference in English and Spanish. You have been asked to participate because we are interested in dialogs within social groups, and the UTEP Computer Science community is such a group.

Procedure: If you agree to participate, you will 1) fill out a short demographic survey, 2) be paired with another study subject, 3) wear a microphone headset, 4) converse with the other person in another room across a glass window, 5) do so for 5-10 minutes, while 6) being audio- and video recorded.

Users of the Data: The data collected will be used by researchers in UTEP's Interactive Systems group and by researchers at other institutions who sign an agreement to follow all the confidentiality and privacy protections described below.

Uses of the Data: 1) After we collect the dialogs, we will recruit CS students to tag them for topics and other aspects that people might like to search on. 2) The collection of dialogs, together with these tags, will be analyzed here at UTEP's Interactive Systems Group and by researchers at other institutions. In general this analysis will be based on statistical analysis of automatically extracted features, but the researchers may listen to or view some of the data. Based on these analyses, researchers will develop an evaluate algorithms and methods for searching in or otherwise processing social multimedia 3) Excerpts of the data may be used in presentations at research meetings to illustrate these algorithms and methods. 4) Transcripts of small dialog segments may be used in scientific articles describing the algorithms, methods, and findings. 5) If you permit by initialing below, short audio snippets of the data (no video), not to exceed 8 seconds, may be placed on websites as illustrations of common patterns of interaction and how they differ in English and Spanish, or otherwise made publically available for educational purposes.

Risks and Confidentiality: The risks involved in participating in this study relate to the loss of privacy and the potential for embarrassment, in particular, since the taggers will be your peers and maybe your friends. To mitigate these risks, your name will not be associated with the dialog(s) you participate in. Despite this, taggers and other local users of the data may still be able to identify you from the video or your voice. We therefore request that you *please avoid saying anything that might embarrass you or anyone else.* If you inadvertently say something that should not be retained, please immediately make a note, and tell the experimenter after the dialog ends, so that he can delete those segments of the audio and video. The experimenter, at his discretion, may also delete short segments if he thinks there is a

Appendix C: Consent Form

potential privacy risk. All recordings will be stored only on password-protected computers and all backup media in locking file cabinets.

Benefits: As described in the flier, your compensation will be either class credit or a small cash amount.

Source of Funding: UTEP and Dr. Ward are receiving funding from the National Science Foundation to conduct this study.

Withdrawal from the study: Taking part in this study is voluntary. If you choose to take part, you have the right to stop at any time without penalty.

Contacts: You may ask any questions you have now. If you have questions later, you may contact:

Steven Werner	(315) 405-3311	sdwerner@miners.utep.edu
Dr. Nigel Ward	(915) 747-6827	nigel@utep.edu

UTEP's Institutional Review Board (IRB) (915) 747-8841 irb.orsp@utep.edu

Agreement:

I have read each page of this consent form, and I voluntarily agree to participate in this study. I will get a copy of this consent form now and can get information on results of the study later, if I wish.

Participant Name: _____ Date: _____

Participant Signature: _____ Time: _____

Consent form explained/witnessed by:

Printed name: _____ Date: _____

Signature: _____ Time: _____

I hereby give permission for short snippets of the audio recordings to be used for educational purposes as described above.

_____ (initial if you agree)

Appendix D: Verbal Instructions for the Dialog Participants

Thank you for coming in today. I'll give you a couple minutes to read the consent form, and then we can talk about it.

...

Okay, so we are interested in social dialog. As it said, we'll be recording you today, and then using those recordings for research. Do you have any questions?

...

Okay, the critical thing here is the privacy issue. If you're okay with the protections we're planning, could you please sign the form?

...

And if you're okay with it, please also initial to allow Educational Uses of short audio clips.

...

Now let's talk a little about topics. Basically you can talk about anything you want. But we're hoping that the recordings include at least some things that other computer science students could listen to and get something out of. As we said in the flier, we think that in social media in the future, you'll be able to search through friends' conversations to find useful things. Assuming they give permission, of course, but that's a separate issue.

So you might talk about things like this semester's classes, and how you're coping, and strategies for success in this major, and career plans or internship experiences, etc. But CS students are interested in lots of things, so it doesn't all have to be serious; if you talk a little about where to eat on campus, or where to park or good movies you've seen lately, et cetera, et cetera, that's great too.

Okay, so we're almost ready. We'd like you to talk for 10 minutes or so; less is fine, a little more would be fine too.

In the recording rooms there are pencil and paper available, so if you say something that ought to be deleted, please note down the time and I'll go in later and take it out.

Okay, let's go set things up.

Appendix E: Metadata

```
conversation_title, dyadic_relationship
      left_speaker_number, left_speaker_gender,
left_speaker_age_range, left_speaker_classification,
left_speaker_native_language{, left_speaker_alternate_languages}
      right_speaker_number, right_speaker_gender,
right_speaker_age_range, right_speaker_classification,
right_speaker_native_language{, right_speaker_alternate_languages}

utep_000, friends
      1, male, 26-30, senior, english
      2, female, 23-25, junior, english
utep_001, friends
      3, female, 23-25, senior, spanish, english
      4, male, 26-30, graduate, marathi, hindu, english
utep_002, acquaintances
      5, male, 23-25, senior, spanish, english
      6, male, >40, other, english
utep_003, friends
      7, male, 20-22, senior, english, spanish
      8, male, 20-22, junior, english, spanish
utep_004, classmates
      9, female, 20-22, junior, english
      10, male, 23-25, junior, english
utep_005, acquaintances
      11, male, 23-25, senior, english
      12, male, 26-30, graduate, english
utep_006, friends
      13, male, 20-22, sophomore, spanish, english
      14, male, 23-25, sophomore, english, spanish
utep_007, classmates
      15, male, <20, freshman, english
      16, male, 20-22, freshman, spanish, english
utep_008, friends
      17, male, <20, sophomore, english
      18, male, 20-22, sophomore, english, spanish
utep_009, friends
      19, male, 20-22, senior, spanish, english
      11, male, 23-25, junior, english
utep_010, acquaintances
      20, male, 26-30, senior, english, spanish
      21, male, 22-25, senior, english, spanish
utep_011, classmates
      11, male, 23-25, senior, english
      21, male, 22-25, senior, english, spanish
utep_012, classmates
      19, male, 20-22, senior, spanish, english
      21, male, 22-25, senior, english, spanish
utep_013, classmates
      22, male, 20-22, senior, english
      23, male, 20-22, senior, english, spanish
utep_014, none
      24, male, 22-25, senior, english, spanish
```

Appendix E: Metadata

```
      25, female, >40, junior, spanish, english
utep_015, classmates
      26, female, <20, freshman, spanish, english
      27, female, <20, freshman, spanish, english
utep_016, none
      23, male, 20-22, senior, english, spanish
      28, female, 20-22, senior, english, spanish
utep_017, none
      29, male, 20-22, senior, english, spanish
      30, female, 20-22, junior, english
utep_018, classmates
      31, male, 20-22, freshman, english
      32, female, 23-25, senior, english
utep_019, classmates
      33, males, 26-30, graduate, spanish, english
      34, males, 31-40, graduate, spanish, english
utep_020S, none
      35, male, 20-22, junior, spanish, english
      36, male, 20-22, senior, spanish, english
utep_021, friends
      30, female, 20-22, junior, english
      2, female, 23-25, junior, english
utep_022S, classmates
      37, female, <20, junior, spanish, english
      38, female, 23-25, junior, english, spanish
utep_023, acquaintances
      39, male, 20-22, senior, spanish, english
      19, male, 20-22, senior, spanish, english
utep_024, acquaintances
      40, male, 20-22, senior, english
      41, male, 20-22, senior, english, french, german
utep_025, friends
      42, male, 20-22, senior, spanish, english
      43, female, 20-22, senior, spanish, english
utep_026, none
      44, male, 23-25, senior, english, spanish
      45, male, 20-22, senior, spanish, english
utep_027, aquaintances
      46, female, 26-30, junior, spanish, english
      43, female, 20-22, senior, spanish, english


S indicates the Spanish-language dialogs, dialogs 20 and 22
Dialogs 19 and 23-27 were the testset dialogs
All others (dialogs 0-18 and 21) were the training set dialogs
```

Appendix F: Annotator Training Overview

Annotator Training Overview

May 6, 2013, Nigel Ward

Thank them for helping us with our research.

Review the compensation ($10 per hour), and projected time required (20 hours).

Briefly review our vision of search in social multimedia.

Explain why we need their help (to tag similar sets, so we can tune our algorithms).

Step 1: Familiarization with the Corpus and Elan

Show them 20 seconds of sample video (dialog 000). Explain the corpus.

Show them how Elan works, show them how to annotate a couple regions.

Let them try it out for themselves.

Give them the How To Tag Using Elan document, and have them try out all the commands.

Step 2: Familiarization with the Task

Give them the Annotators Guide and ask them to read the Guidelines section. Give them lots of time, then answer any questions.

Have them label the first 5 minutes of one of the nicest dialogs (for labler 2 use 004, labeler 3 006, 4 008, and 5 012).

Then review the tags they came up with, and discuss them with them. Call Nigel at x6827 to come discuss also, if he's around. Unless they're way out of line or misunderstanding something fundamental, just praise their decisions and listen to their justifications.

Copy the .eaf file they created somewhere else, so they have a clean start.

Then set them to work, according to the steps in the Procedure section of the Annotators Guide.

This file is annotator-training-overview.txt in /home/users/nigel/papers/mediaeval/instructions/ .

Social Multimedia Search May 6, 2013, Nigel Ward

Annotators Guide

Thank you for coming in to help us with our research.

We are building fully automated search tools for audio, but need human help at this stage, to set the targets for how the algorithms should behave. In particular, we are gathering judgments of what things in the recordings are *similar*. The idea is that a listener may be browsing and hear something interesting to him, and then want to be able to find more things "similar to" that. But there's no hard definition of similarity, so we need your judgments.

Tagging Guidelines

You may have experience with tagging from twitter, flickr, tumblr etc., which may be helpful, but since this is a new kind of application, new ways of tagging may be more appropriate.

Tags should be useful. If you're not sure whether to invent/adopt a tag, please think about who it might be useful for. Something that would be useful to you in future when browsing data like this would certainly make a good tag. However there are other potential searchers, with various needs, that you might consider. You might think of fellow students looking for anything interesting, or fellow students deciding what classes to take next semester, or students wondering if anyone else is having trouble with stat and if so how they're dealing with it, freshmen deciding whether to major in computer science (or whether to do it at UTEP), or maybe even outsiders, for example a professor wanting to discover what students are most pleased about, excited about, or worried about in their major, or someone out of town wondering what kinds of games and movies are popular in El Paso.

Tags may be indirectly useful. In particular: 1. Tags for things no one would want to search on, such as "boring repetitive complaints," can still be useful, as a way to help a search engine learn what sorts of things to exclude from the results. 2. Tags that might be useful only in conjunction other tags are still worth including. For example, a tag for "serious technical talk" might be useful together with "game development platforms," or a tag for "talk about experiences a semester or more ago" might be useful together with "discussing what non-CS classes to take".

Ephemeral tags are okay. The data you are listening to were recorded weeks ago, and some of the topics are by now irrelevant, for example the bomb scare. However in future we might imagine a twitter-like scenario, in which searchers want to find things said within the past hour on a specific topic. So it's okay to have tags for ephemeral things.

Social tags are encouraged. While the UTEP CS community is interested in serious things, like internships, projects and technology, people are also interested in lighter topics, like food on campus, morning traffic, finding daycare for children, funny stories, and so on, and it's fine to tag such things too.

Overlapping tags are fine. Sometimes you will probably want to assign multiple tags. For example, if someone is talking about CS 4375 and complains about the homework and the other person explains a

trick that helped them complete it, that may be labeled "CS 3195"and "homework" and "complaints" and "explaining technology". Tags may overlap fully or partially. For example, if the discussion of homework is for only a few seconds in a larger discussion of that class, it is best if the homework tag covers just that subregion.

Untagged content is fine. Unlike existing social media, people in these conversations are just talking to each other, not designing things for third-parties to view. So some of the things they say are of no general interest, and so large stretches of the dialogs may need no tags at all.

Tag times can be approximate. Tagged regions can be of any length, from a couple of seconds to a few minutes. Often the starts and ends of regions are ambiguous, especially when a topic slowly dies out. Region starts should be marked accurately to within about a second, however region ends will, realistically, probably be less accurate.

Procedure

Also see How to Tag Social Speech using Elan

Step 1. [45 minutes] Listen to the sample audio and learn how to use Elan.

Step 2. [45 minutes] Practice choosing and assigning tags. Then discuss with the experimenter the tags you chose, and any questions or suggestions you have.

Step 3. [90 minutes] Identify some potential tags. Listen to an hour of the dialogs, noting down some ideas for tags you may like to use. Based on your annotator ID, please do this for the following subset: annotator 1 (Nigel): 003-008; annotator 2: 006 and 009-012; annotator 3: 008, 013-015 and 017; annotator 4: 012, 017-18, 021, and 004; annotator 5: 004, 010, 012, 014, 018

Step 4. [90 minutes] Choose your tags. In an editor, create a file of your tag ideas as a big list. You probably will have several dozen. Sort them into similar groups. Review them and tidy them up. For example:

- if you discover two that are different names for the same thing, combine them.
- if you discover a tag that relates to only one segment, remove it, for example, #complaints-about-cs-3331-professors-haircut
- if you discover a tag that is too probably too general to be useful for searchers, remove it, for example, #classes.
- if in doubt, refer back to the Tagging Guidelines above, or discuss with the experimenter (bearing in mind that he has not much more experience with this than you).

We honestly don't know how many tags are appropriate for this data set, if there are more than 50 it will be hard to remember them all as you tag, and if there are less than 30, you may have too few to tag most of the content, or else they may be too general to be useful.

Step 5. [10 minutes] Input your tags. In Elan, use Edit Controlled Vocabularies.

Step 6. [10-20 hours] Assign the tags. Do this for all dialogs: 000-018 and 021. Remember that the goal is to tag things to support search, so don't do this mechanically:

Appendix G: Annotators Guide

- For example, even if you have a tag #gradschool, you may not want to use it for someone saying "my sister's finished graduate school and is now a nurse practitioner," since CS students interested in learning about graduate schools would probably find no useful information in that.
- For example, if you have a tag #favorite-professors, and the topic is a favorite TA, then go ahead and use that tag. Remember that it's all about similarity, not precise matches.

If people are talking slowly, you may be able to tag as you listen. Otherwise it may be better to first listen once and jot down tag ideas and rough times on a sheet of paper, and then do the data entry in a second pass.

The first dialog will probably take you over an hour to label just 10 minutes, but soon you'll get faster. Probably do no more than two at a sitting, and the work is tiring and you may lose attention if you don't take breaks.

As you work, please ask us any questions you have, and please let us hear your thoughts about:

- how we might make tagging easier or more realistic,
- what functionality to include in search engines for social multimedia
- what privacy concerns people might feel if data like this were available to their social group
- anything else.

Step 7. [30 minutes] Assign a weight to each tag. Probably by this point you'll be feeling that some of the tags were really useful, and some less so. For each tag, please assign it a weight, from 0 to 3.

3 – clearly useful tag. A tag that a system should handle well; one that occurs often in the data, but is not over-frequent; one that covers some high-quality content; one that would be directly useful to searchers; one with a coherent meaning that you could consistently apply.
2 – useful, but not as great as a level-3 tag.
1 – possibly useful
0 - pretty much worthless (for example, a tag that you used only once)

Step 8. [20 minutes] Fill out the privacy questionnaire, and discuss your tags and the whole experience with the experimenter.

You're now done. Thank you!

Appendix G: Annotators Guide

in nigel/papers/mediaeval/instructions/annotator-instructions.docx;
also see nigel/papers/mediaeval/instructions/annotator-training-overview.txt
and see isg/speech/social/tagging-howto.docx

Appendix H: How to Tags Social Speech using Elan

How to Tag Social Speech using Elan

Version 1, May 5, 2013, Nigel Ward

This document describes the mechanics of tagging, organized by the steps listed in the Annotators Guide document.

Steps 1-4: Familiarization and Exploration

To create an annotations file:

- Open Elan
- menu: File, New
- navigate to the social/video directory, set it to show all file types, then pick a file
- click ">>" to move it into the workspace, then click "OK"

You will see the movie. Click play and verify that the sound works.

Useful Elan Commands and Shortcuts:

- click to place the current timepoint (vertical cursor)
- cntl-space to play from the current timepoint
- drag mouse to select region
- shift-space to play selected region

- drag "rate" slider to select rate (130 or 140 is good)
- rightclick in the second-timeline and "zoom" to 10% to compress signal

- double-click (or alt-n) to add a tag to the current region
- double-click or control-enter to close and save a tag

- if you want to add a tag that overlaps an existing tag, put the new one in a different tier

Step 5: Tagset Entry

Create a directory for your tags under "isg/speech/social", e.g prunellas-eaf (if your name is Prunella).

Open up Elan, on any movie file

menu: Edit, Edit Controlled Vocabularies
give your tagset a name, e.g. prunella-tags
enter all the tags (in the entry value box) and "add" each one
close to save it; don't exit Elan yet

menu: Type, Add New Linguistic Type

Appendix H: How to Tags Social Speech using Elan

add, type name = prunella-type (if your name is prunella)
"use-controlled vocabulary" to select the tagset you created a moment ago
"Add to save"

menu: Tier, Add New Tier
enter a tier name, topics1
select as "linguistic type" the type you created a minute ago
add
repeat the sequence twice to create topics2 and topics3 tiers

create a test tag:
drag over a region of time.
double-click or alt-n to edit.
select the desired tag

Menu: File, Save; save the file in prunellas-eaf, using the same name as that of the video,
For example, save utep999.m4v as utep999. The .eaf extension will automatically be written.
Verify that the eaf file you wrote exists and is contains your tagged region plus your whole list of tags.

Now that you've saved a tag, you don't want to overwrite it. So in future, when you want to open this file, use "Open" not "New", and just click on the eaf file. Try it:
Exit elan
Open your file again, see if the tiers are still there, and your test tag.

Step 6: Tagging

Label the whole thing, saving periodically

To go on to the next file
New, then load the video file
Tier, then import the tier from your previous .eaf file (to get your tiers and tagset)
Save, then set the save destination to be the appropriate .eaf filename for this movie
then go to work labeling.

Step 7: Assigning Weights

Print out any of the .eaf files, find the tags at the bottom, and pencil in a weight for each, from 0 to 3.

Alternatively, ask the experimenter to process the eaf files, then edit the numbers in the resulting "flattened" file, replacing the default 2s with the weights you like.

in isg/speech/social/tagging-howto.docx
see also nigel/papers/mediaeval/instructions/annotator-instructions.docx

Appendix I: slimcrunch.fss, the Feature Set Specification

```
#slimcrunch.fss
# This file specifies which prosodic features to gather up,
#  at various temporal offsets,
# Nigel Ward, UTEP, January 2012

### Past Features
vo   -50 to     0 self
vo  -100 to   -50 self
vo  -200 to  -100 self
vo  -300 to  -200 self
vo  -400 to  -300 self
vo  -800 to  -400 self
vo -1600 to  -800 self
vo -3200 to -1600 self

vo  -200 to     0 inte
vo  -400 to  -200 inte
vo  -800 to  -400 inte
vo -1600 to  -800 inte
vo -3200 to -1600 inte


ph   -50 to     0 self
ph  -100 to   -50 self
ph  -200 to  -100 self
ph  -400 to  -200 self
ph  -800 to  -400 self

ph  -200 to     0 inte
ph  -400 to  -200 inte
ph  -800 to  -400 inte


pr   -50 to     0 self
pr  -100 to   -50 self
pr  -200 to  -100 self
pr  -400 to  -200 self
pr  -800 to  -400 self

pr  -200 to     0 inte
pr  -400 to  -200 inte
pr  -800 to  -400 inte


sr   -50 to     0 self
sr  -100 to   -50 self
sr  -200 to  -100 self
sr  -400 to  -200 self
sr  -800 to  -400 self
sr -1600 to  -800 self

sr  -200 to     0 inte
sr  -400 to  -200 inte
sr  -800 to  -400 inte
```

Appendix I: slimcrunch.fss, the Feature Set Specification

```
sr -1600 to  -800 inte


###  Future Features

vo     0 to    50 self
vo    50 to   100 self
vo   100 to   200 self
vo   100 to   300 self
vo   300 to   400 self
vo   400 to   800 self
vo   800 to  1600 self
vo  1600 to  3200 self

vo     0 to   200 inte
vo   200 to   400 inte
vo   400 to   800 inte
vo   800 to  1600 inte
vo  1600 to  3200 inte


ph     0 to    50 self
ph    50 to   100 self
ph   100 to   200 self
ph   200 to   400 self
ph   400 to   800 self

ph     0 to   200 inte
ph   200 to   400 inte
ph   400 to   800 inte


pr     0 to    50 self
pr    50 to   100 self
pr   100 to   200 self
pr   200 to   400 self
pr   400 to   800 self

pr     0 to   200 inte
pr   200 to   400 inte
pr   400 to   800 inte


sr     0 to    50 self
sr    50 to   100 self
sr   100 to   200 self
sr   200 to   400 self
sr   400 to   800 self
sr   800 to  1600 self

sr     0 to   200 inte
sr   200 to   400 inte
sr   400 to   800 inte
sr   800 to  1600 inte
```

Appendix J: Guide to Interpreting the Metrics

Similar Segments of Social Speech Task

GUIDE TO INTERPRETING THE METRICS

Nigel Ward, July 30, 2013

The basic evaluation philosophy for this task is described in the workshop task-overview paper: The Similar Segments in Social Speech Task. In short, the idea is to use a simulation of user behavior to indicate how useful any similar-region-suggesting system would be, and a single overall quality metric is proposed. Please refer to that paper for the description.

These notes discuss the realism and stability of the main measures, explain the normalization factors, and describe all the various measures.

1. Validation, Stability, and Region-Length Effects

To validate the computation of these metrics, I created various small test sets, and verified that the computations worked as intended. I also tested both a random baseline and a "clever" reference system, described below, on both the trainingset queries and the testset queries. I also varied a few of the evaluation parameters.

Across these experiments, the only unexpected influence on importance was region length. For example, the random algorithm did surprisingly well on the trainingset data, and this was mostly due to the fact that many of the tagged regions were very long (as discussed in the Annotation Notes document), and some of the tagsets were quite large. For example, a tag like "entertainment" could apply to large fraction of the entire corpus. Such general tags were not envisaged when the task was designed, however they do not seem unrealistic or inappropriate. Longer regions appear to work slightly to the advantage of random algorithm, in that points selected at random at tend to fall more in longer regions than in short regions, and that both the coverage and benefit values are increased to the extent that longer content is found. Although this doesn't seem to be entirely inappropriate, it reduces the incentive for systems to find lots of similar regions, and instead lets them score well by just finding one, very long region. Fortunately there were fewer such very-long regions in the testset, so this issue is moot.

The prevalence of longer regions did lead us to abandon the "scan-back" action included in the original user-behavior simulation. This modeled the idea that a user encountering a jump-in point which took her to the middle of a useful region might then seek back to find the start of that region and listen from the start. For long regions and/or regions with diffuse content, like "entertainment", this seemed unrealistic; a user would probably just listen from the jump-in point to the end, and then go on to the next jump-in point, to find reasonable content with less hassle. The user-behavior simulator was simplified accordingly.

Appendix J: Guide to Interpreting the Metrics

2. Adjustment Factors

To estimate the maximum achievable performance, I built a "clever" reference algorithm that used the information in other tagsets. Specifically, given a query region, the algorithm looks through all regions in all other tagsets and finds the region which overlaps the query most closely. It then returns, as jump-in points, the onsets of all other regions in the same tagset as this strongly-overlapping region. If there are fewer than 20 such regions, it then does a new scan to find the the next most overlapping region, and uses the regions in its tagset to generate more jump-in points. It continues until 20 jump-in points have been generated or until there are no regions whose amount of overlap is 40% or more of the sum of the durations of the overlapping pair of regions.

Thus, this algorithm exploits the information provided by other taggers. Of course, no realist similarity system would have access to such information. However this algorithm is useful for estimating the upper bound on system performance. In particular, even on the trainingset, replete with long regions, this algorithm attained an raw F-measure of only .43, clearly a long way from 1.00. Thus, as noted in the task-overview paper, adjustments are needed.

In essence, adjustments are needed because tagsets identify some similar regions but not all. Thus they never let us know for sure that a putative result is {\em not} similar to the query region. Thus the raw measures severely understate the actual utility. As an extreme example, imagine that an annotaator the exact same region with two different tags, for example perhaps 'ai-class' and 'favorite professors'. A query with that region could validly return either other ai-class regions or other favorite- professor regions, but the scoring algorithm will pick one of the two tagsets and use that to evaluate all results returned, meaning that half of them will be counted as false alarms, unjustly. Indeed there are a few such multiply-tagged regions in the data, so this is not hypothetical.

In principle we could overcome these problems by having human judges evaluate, post-hoc, the quality of each jump-in point. This would give us explicit judgments of non-similarity, but unfortunately this is not be affordable. Therefore we continue to rely on the tagsets, but adjust the raw scores upwards.

The adjustments depend on the exact corpus subset and query set. For the testset data, the Searcher Utility Ratio is divided by 0.290 which is the raw score obtained by the clever algorithm. The recall is divided by 0.275, which is the value the clever algorithm would have obtained had it given jump-in points for all queries (not just the 67% it did answer) at the same recall level that it achieved for the ones it did answer (18.4%).

3. Explanation of Measures

While the F-measure is the primary measure of system quality, there are other measures which can be used, especially to help understand the various strengths of the systems, and some of these are included

Appendix J: Guide to Interpreting the Metrics

in the output of score5.py.

The "naive precision" is the fraction of jump-in points that matched a region in the same tagset as a query, without being converted to seconds and without penalties for jump-in points being early or late.

The "average seconds early" and "average seconds late" are reported since it's helpful to know where the jump-in points are falling: it being better for a jump-in point to be before a target-region onset than after, and it being better for the jump-in points to be close to the region onset, rather than further away.

The raw recall figure here is, as described in the task-overview paper, not the traditional fraction of total relevant segments retrieved, but the number of seconds of relevant data that a user (as simulated) could get from these within the 120 second per-query time, divided by the total number of seconds that an ideal system could deliver in that time.

Finally, the F-measure is computed. While the searcher utility ratio by itself is probably the most meaningful measure, it is possible that a system might score very highly on this metric by only generating one jump-in point for one query; something that would not be very useful in most scenarios. Accordingly the recall factor is also incorporated in the final score, combined with utility as an F-measure. However utility is the most important component, so the F-measure is weighted to favor it 9 to 1.

/home/users/nigel/papers/mediaeval/guide-to-metrics.txt

Appendix K: Baselines

```
Similar Segments of Social Speech Task

                        BASELINES ON THE TESTSET

Nigel Ward, July 17, 2013



BASELINE 1: RANDOM GUESSING

['../python/score5.py', 'testset/testset-queries.txt',
'../tagsets/testset/all-tagsets-pruned-SECRET.txt', 'testset/random-
guesses.txt']


SUMMARY:
  processed 21 queries
   of which 0 lacked answers entirely
   the answers examined (within 120 sec. per query) included:
     195 false alarms and 15  total hits
     4 successful and exact-or-early jump-in points
       averaging 2.4 seconds early
     11 successful but late jump-in points
       averaging 32.5 seconds late
   naivePrecision =  7% = 15 / 210

  raw recall = 11% = 223 / 2020 seconds
  raw searcher utility ratio: 0.125  = 10.6 / 85.4
    (average cost  / average benefit (both per query, both in seconds))
  Normalized Searcher Utility Ratio: 0.430
  Normalized Recall: 0.402
  F-measure 0.43



BASELINE 2: THE "CLEVER ALGORITHM" (as described in the Guide to
Interpreting the Metrics)

['../python/score5.py', 'testset/testset-queries.txt',
'../tagsets/testset/all-tagsets-pruned-SECRET.txt', 'testset/inferred-
answers.txt']


SUMMARY:
  processed 21 queries
   of which 7 lacked answers entirely
   the answers examined (within 120 sec. per query) included:
     112 false alarms and 11  total hits
     5 successful and exact-or-early jump-in points
       averaging 2.3 seconds early
     6 successful but late jump-in points
       averaging 42.5 seconds late
   naivePrecision =  9% = 11 / 123
```

Appendix K: Baselines

```
  raw recall = 18% = 371 / 2020 seconds
  raw searcher utility ratio: 0.290  = 17.7 / 61.0
    (average cost  / average benefit (both per query, both in seconds))
  Normalized Searcher Utility Ratio: 1.001
  Normalized Recall: 0.669
  F-measure 0.95
```

/home/research/isg/speech/social/baselines/baselines.txt

Appendix L: score5.py, the scoring program

```
# score5.py     Python 2.5 script (fails on Python 2.4)

# /home/research/isg/speech/social/python/score4.py

# This script scores the quality of matches
#   for the Simlar Segments in Social Seach MediaEval 2013 Task
# Nigel Ward, University of Texas at El Paso, March 2013

# Version 3, May 13, 2013: changed to omit query weighting
# Version 4, July 2013: added more information to the performance summary
# Changed so that the hunt-back-to-start cost is estimated at 50% of
#   the distance, instead of 100%, assuming people jump back 10 seconds
#   at a time, they listen for 4 seconds each time, and jump overhead is
1 sec.
# Then changes so that there is no hunt-back-to-start behavior
#  (plausible especially when the regions are very long)
# Changed to add the 'naive precision' and the recall

# Given a query, this looks up the answerset for that query
#   and evaluates its quality by comparing it to the reference similarity
set.
# It implements the algorithm descrbed at
#    http://www.cs.utep.edu/nigel/ssss/faq.html
# The output is, for each query,
#   a description of how well that query was handled.
#   and the Searcher Utility Ratio.
# It also prints out the average Ratio over all queries.

# invoke with   python score3.py queries.txt sss.txt answers.txt

# in ../baseline:
# python ../python/score4.py all-possible-queries2.txt all-similarity-
sets-clean2.txt all-infered-answers.txt; gives naive precision 9%, recall
27%, utility ratio .31, f-measure of 0.29
# ditto on random-guesses.txt: 6%, 30%, 24%, 0.27


# The inputs are three files:
# The list of queries, provided by the organizers
# The lists of answers, that is, list of jump-in points inferred for each
#   of the queries, as generated by a participant system.
# The reference list of similar-segment sets, provided by the organizers.
#   For the final evaluation set this will not be revealed to the
participants.

# Note that all the constants are designed to reflect plausible
# searcher behavior, with the exception of the noResults penalty.
# Real searchers would probably prefer to have zero results than one
# incorrect result, but for purposes of the evaluation, we don't want
# to enable systems to get high scores by only the one or two queries
# they're confident they can do well on.

import sys
```

Appendix L: score5.py, the scoring program

```
# constants
searchTimePerQuery = 120    # seconds
maxLeftOffset = 5     # seconds
minRightOffset = 3    # seconds; was 1, changed in version 4
falsePositivePenalty = 8 # seconds
#searchBackPenaltyRatio = 2  # considered making it lower

def skippable(line):
  if len(line) == 0:
     return True
  if line[0] == '#':
     return True
  if line.isspace():
     return True
  return False

# Here string1 and string2 both have format: filename  start  end
# Returns true if they represent the same region
#   (regardless of whitespace variation and leading/trailing zeros)
def sameRegion(string1, string2):
   triple1 = string1.split()
   triple2 = string2.split()
   return (triple1[0] == triple2[0] and float(triple1[1]) ==
float(triple2[1]) and float(triple1[2]) == float(triple2[2]))


# Return all result-regions corresponding to the specified query, by
#  scaning through the answers file to find a line prefixed by "input:"
#  that contains the exact same filename, start time and end time,
#  and then snatching up all lines after that (up to the next "input:")
def findResultSet(query, lines):
   jumpinset = []
   nlines = len(lines)
   for i in range(nlines):
       if skippable(lines[i]):
          continue
       #print "  answerfile has: ", lines[i],
       triple = (lines[i]).partition(" ")
       firstToken = triple[0]
       if firstToken != 'input:':
           continue
       remainingTokens = triple[2].strip()
       if not sameRegion(query,remainingTokens):
           continue
       # we've found the corresponding set; collect the relevant lines
       i = i + 1
       while i < nlines:
           line = lines[i].strip()
           triple = lines[i].partition(" ")
           firstToken = triple[0]
           if skippable(line) or firstToken == 'input:':
              print '  in answerfile, jumpinset is: ', jumpinset
              return jumpinset  # all done
           jumpinset.append(line.strip())
```

Appendix L: score5.py, the scoring program

```
            i = i + 1
    return jumpinset   # which will be null

def findSimilarSet(query, sslines):
    similarset = []
    nlines = len(sslines)
    for i in range(nlines):
        ssline = sslines[i]
        if skippable(ssline):
          continue
        if ssline.find("set") > -1:
          continue
        if not sameRegion(sslines[i].strip(),query):
          continue
        # search back collecting all lines until we see a 'set' token
        j = i - 1
        while j > 0:
            ssline = sslines[j].strip()
            if skippable(ssline):
                break
            tokens = ssline.split()
            if tokens[0] == 'set:':
               break
            similarset.append(ssline)
            j = j - 1
        # search forward analogously
        k = i + 1
        while k < nlines:
            ssline = sslines[k].strip()
            if skippable(ssline):
                break
            tokens = ssline.split()
            if tokens[0] == 'set:':
               break
            similarset.append(ssline)
            k = k + 1
        if similarset == []:
          print "warning, no similar regions in reference tagsets for",
query
        return similarset
    print "warning, no region in reference tagset file matches", query


# A query is a string: a filename, a start time, and an end time
# We seek a corresponding answer set in the answers file.
# Any answersets that don't correspond to a query are ignored.
# If no corresponding answer set is found, the benefit for this
#   query is zero, and the noResults penalty is applied
# We also seek a corresponding similarity set in the ss file.
# Finally we call simulateUser to estimate the value of this answer set
#  as a set of jump-in points leading to discovery of the similarity set.
def processQuery (query):
    global nqueries
    global cumulativePotential
```

Appendix L: score5.py, the scoring program

```
    nqueries = nqueries + 1
    print "\nProcessing query", query
    similarset = findSimilarSet(query, sslines)  # reference tagset
    resultset =  findResultSet(query, answerlines) # system guesses
    unmaxedPotentialValue = totalSetDuration(similarset)
    potentialValue = min(searchTimePerQuery, unmaxedPotentialValue)
    cumulativePotential = cumulativePotential + potentialValue
    simulateUser(query, resultset, similarset, potentialValue)

def totalSetDuration(similarset):
   sum = 0
   for region in similarset:
     triple = region.split()
     start = float(triple[1])
     end = float(triple[2])
     sum = sum + (end - start)
   return sum


# simulateUser For each putative answer in the set, determine its
# value and cost, and keep processing putative answers until the total
# cost for this query exceeds 120 seconds

#  secondswo shared by the daughter function ProcessResult
thisQueryCost = 0
thisQueryValue = 0

def simulateUser(query, resultset, tagset, potential):
    global cumulativeCost
    global cumulativeBenefit
    global cumulativeNoPutative
    #  globals shared by the daughter function ProcessResult
    global thisQueryCost
    global thisQueryValue

    if resultset == []:
        print 'no putative results for', query
        cumulativeNoPutative = cumulativeNoPutative  + 1
        return
    thisQueryCost = 0
    thisQueryValue = 0
    liveTagSet = tagset  # regions not already jumped-into by an answer
    for result in resultset:
       print '   processing result: ', result
       liveTagSet = processResult(result, liveTagSet)
       if thisQueryCost >= searchTimePerQuery:
         # if times up, stop processing results
           break

    thisQueryValue = min(thisQueryValue, searchTimePerQuery)
    print "for this query, cost is", thisQueryCost, 'and benefit',
thisQueryValue, "of a possible", potential
    print "Searcher Utility Ratio for query '", query, "' is",
    print '%.2f' % (thisQueryValue / thisQueryCost)
```

Appendix L: score5.py, the scoring program

```
    cumulativeCost = cumulativeCost + thisQueryCost
    cumulativeBenefit = cumulativeBenefit + thisQueryValue


# There are three cases:
#  Case 1, if the answer is within an unused similar segment,
#    and no later than 1 second before the end, then
#    value = duration of that segment
#    cost = duration of that segment + twice the distance from
#         the jump-in point to the start of the segment
#    and mark that segment as used up
#  Case 2, if the answer is no more than 5 seconds before an
#     unused similar segment
#    value = duration of that segment
#    cost = time distance from jump-in point to segment onset
#    and mark that segment as used up
#  Case 3, if none of the first two applies, it's a false alarm
#    cost = 8 seconds
#    value = 0 seconds
def processResult(result,ltset):
    global thisQueryCost
    global thisQueryValue

    global cumulativeFalseAlarms
    global cumulativeEarlyHits
    global cumulativeLateHits
    global cumulativeEarliness
    global cumulativeLateness

    triple = result.split()
    rfilename = triple[0]
    rtimepoint = float(triple[1])
    # find the first matching item in the liveTagSet
    if ltset == None:
       # no (remaining live) regions in similarity set, stop scoring
answers
       # thus there's no penalty if the system returns more answers than
       # there are regions, as long as the correct answers are at the top
      thisQueryCost = thisQueryCost + falsePositivePenalty
      cumulativeFalseAlarms = cumulativeFalseAlarms + 1
      print '     Case 3: no similar regions (left to try to) match this
jump-in point'
      return ltset
    for similarRegion in ltset:
        triple = similarRegion.split()
        sfilename = triple[0]
        sstartpoint = float(triple[1])
        sendpoint = float(triple[2])
        if sfilename != rfilename:
            continue  # keep looking to find a similar region
        if rtimepoint > sstartpoint and rtimepoint <= sendpoint -
minRightOffset:
            # Case 1
            #value = sendpoint - sstartpoint
```

Appendix L: score5.py, the scoring program

```
            #cost = value + searchBackPenaltyRatio * (rtimepoint -
sstartpoint)
            value  = sendpoint - rtimepoint
            cost = value
            thisQueryCost = thisQueryCost + cost
            thisQueryValue = thisQueryValue + value
            cumulativeLateHits = cumulativeLateHits + 1
            cumulativeLateness = cumulativeLateness + (rtimepoint -
sstartpoint)
            print "     Case 1: jump-in point within similar region"
            ltset.remove(similarRegion)
            return ltset
        if rtimepoint >= sstartpoint - maxLeftOffset and rtimepoint <=
sstartpoint:
            # Case 2
            value = sendpoint - sstartpoint
            cost = value + (sstartpoint - rtimepoint)
            thisQueryCost = thisQueryCost + cost
            thisQueryValue = thisQueryValue + value
            cumulativeEarlyHits = cumulativeEarlyHits + 1
            cumulativeEarliness = cumulativeEarliness + (sstartpoint -
rtimepoint)
            print "     Case 2: jump-in point preceeds similar region"
            ltset.remove(similarRegion)
            return ltset

    # Case 3: we've scanned all similar regions but found no match
    thisQueryCost = thisQueryCost + falsePositivePenalty
    cumulativeFalseAlarms = cumulativeFalseAlarms + 1
    print '     Case 3: no similar regions match this jump-in point'
    return ltset


# ---- main ----

nargs = len(sys.argv) - 1
print sys.argv
print nargs
if nargs != 3:
   print "Score Error: expected 3 arguments, got", nargs
   print "invoke with 'python score4.py queryfile tagsetfile answerfile'"
queryfile = sys.argv[1]
ssfile = sys.argv[2]  # similarity-sets file
answerfile = sys.argv[3]

qfp = open(queryfile, 'r')
sfp = open(ssfile, 'r')
afp = open(answerfile, 'r')
ofp = open('performance.txt', 'w')

queries = qfp.readlines()
answerlines = afp.readlines()
sslines = sfp.readlines()
```

Appendix L: score5.py, the scoring program

```
nqueries = 0
cumulativePotential = 0
cumulativeCost = 0
cumulativeBenefit = 0
cumulativeNoPutative = 0
cumulativeFalseAlarms = 0
cumulativeEarlyHits = 0
cumulativeLateHits = 0
cumulativeEarliness = 0.0
cumulativeLateness = 0.0

for query in queries:
   if skippable(query):
      continue
   processQuery(query.strip())
ofp.write('all done processing queries')

totalHits = cumulativeEarlyHits + cumulativeLateHits
totalGuesses = totalHits + cumulativeFalseAlarms
naivePrecision = 1.0 * totalHits / totalGuesses
recall = cumulativeBenefit / cumulativePotential

print '\nSUMMARY:'
print '  processed', nqueries, 'queries'
print '   of which', cumulativeNoPutative, 'lacked answers entirely'
print '   the answers examined (within', searchTimePerQuery, 'sec. per
query) included:'
print '    ', cumulativeFalseAlarms, 'false alarms and', totalHits, "
total hits"
print '    ', cumulativeEarlyHits, 'successful and exact-or-early jump-in
points'
if cumulativeEarlyHits > 0:
  print '       averaging %.1f' % (cumulativeEarliness /
cumulativeEarlyHits), 'seconds early'
print '    ', cumulativeLateHits, 'successful but late jump-in points'
if cumulativeLateHits > 0:
  print '       averaging %.1f' % (cumulativeLateness /
cumulativeLateHits), 'seconds late'
print '   naivePrecision = %2.0f%%' % (naivePrecision * 100), "=",
totalHits, "/", totalGuesses

print ' '
print '  raw recall = %2.0f%%' % (recall * 100), "=",
int(cumulativeBenefit), "/", int(cumulativePotential), "seconds"
searcherUtilityRatio = (cumulativeBenefit / cumulativeCost)
print '  raw searcher utility ratio: %.3f' % searcherUtilityRatio, " =
%.1f" % (cumulativeBenefit / nqueries * 1.00) , "/ %.1f" %
(cumulativeCost / nqueries * 1.00)
print '    (average cost  / average benefit (both per query, both in
seconds))'
normalizedSUR =  searcherUtilityRatio / 0.290
normalizedR = recall / 0.275

print '  Normalized Searcher Utility Ratio: %0.3f ' % normalizedSUR
```

Appendix L: score5.py, the scoring program

```
print '  Normalized Recall: %0.3f ' % normalizedR

print "  F-measure %.2f" % ((10 * normalizedSUR * normalizedR) /
(normalizedSUR + 9*normalizedR))
```