Technologies That Make You Smile: Adding Humor to Text-Based Applications

Rada Mihalcea, *University of North Texas*

Carlo Strapparava, *Istituto per la ricerca scientifica e Tecnologica*

Natural language's creative genres are traditionally considered to be outside the scope of computational modeling. Computational linguists have paid little attention to humor in particular because it is puzzling by nature. However, given the importance of humor in our daily lives and computers in our work and entertainment, studies related to computational humor will become increasingly significant in fields such as human-computer interaction, intelligent interactive entertainment, and computer-assisted education.

Humor is essential for interpersonal communication, but research often neglects the topic. Computational approaches can be successfully applied to the recognition and use of verbally expressed humor.

Previous work in computational humor has focused mainly on humor generation,[1,2] and little research has addressed developing systems for automatic humor recognition[3] (see the "Related Work on Computational Humor" sidebar). This is not surprising because, computationally, humor recognition appears to be significantly more subtle and difficult than humor generation. Moreover, the absence of very large collections of humorous texts has hindered the development of systems that use humor in text-based applications. Consequently, few such systems are available.

In this article, we explore computational approaches' applicability to the recognition and use of verbally expressed humor. Particularly, we focus on three important research questions related to this problem: Can we automatically gather large collections of humorous texts? Can we automatically recognize humor in text? And can we automatically insert humorous add-ons into existing applications?

One-liners versus long jokes

Because a deep comprehension of all humor styles is probably too ambitious for existing computational capabilities, we restricted our investigation to one-liners. A *one-liner* is a short sentence with comic effects and an interesting linguistic structure: simple syntax, deliberate use of rhetoric devices (such as alliteration or rhyme), and frequent use of creative language constructions meant to attract the reader's attention. For instance, "I'm not a vegetarian because I love animals, I'm a vegetarian because I hate plants" is an example of a one-liner.

Although longer jokes can have a relatively complex narrative structure, a one-liner must produce the humorous effect in one shot, with few words. This makes one-liners particularly suitable for automatic learning settings because the humor-producing features are guaranteed to be present in the first (and only) sentence.

Web-based bootstrapping of humorous one-liners

Large amounts of training data can potentially make the learning process more accurate and at the same time provide insights into how increasingly larger data sets can affect classification precision. However, we found that manually constructing a very large one-liner data set was problematic because most Web sites or mailing lists that had such jokes did not list more than 50 to 100 one-liners. To tackle this problem, we implemented a Web-based bootstrapping algorithm that could collect numerous one-liners starting with a short seed list, consisting of a few manually identified one-liners.

Figure 1 illustrates the bootstrapping process. Starting with the seed set, the algorithm automatically identifies a list of Web pages that include at

1541-1672/06/$20.00 © 2006 IEEE
Published by the IEEE Computer Society

Related Work in Computational Humor

Although scientific fields such as linguistics[1] and psychology[2] have studied humor relatively well to date, only a little research has addressed the construction of computational humor prototypes.

An early project devised a formal model of semantic and syntactic regularities underlying some simple types of puns.[3] A system called JAPE (Joke Analysis and Production Engine) then exploited the model to automatically generate amusing puns.

HAHAcronym, another humor-generation project, developed a system that automatically generated humorous versions of existing acronyms and produced a new amusing acronym constrained to be a valid vocabulary word.[4] The system achieved a comic effect mainly by exploiting incongruity theories (for example, finding a religious variation for a technical acronym).

Another project focused on humor comprehension, specifically knock-knock jokes, a restricted type of wordplay.[5] The study evaluated to what extent wordplay can be automatically identified in knock-knock jokes and whether such jokes can be reliably recognized from other nonhumorous text. The algorithm was based on automatically extracted structural patterns and heuristics originating from the knock-knock joke's peculiar structure. Although the wordplay recognition gave satisfactory results, the identification of jokes containing such wordplays was significantly more difficult.

Finally, a related project explored humor detection through multimodal analysis of speech and gesture.[6] Researchers measured four features of Jay Leno's monologue on "The Tonight Show with Jay Leno" to identify possible correlations between speech and gesture factors. They could automatically find portions of the monologue during which Leno made large gestures by identifying long pauses in his speech. In practice, this usually corresponds to a point of emphasis and occurs at a joke's conclusion.

References

1. S. Attardo, *Linguistic Theory of Humor*, Mouton de Gruyter, 1994.
2. W. Ruch, "Computers with a Personality? Lessons to Be Learned from Studies of the Psychology of Humor," *Proc. April Fools Day Workshop on Computational Humour,* ITC-irst, 2002, pp. 57–70.
3. K. Binsted and G. Ritchie, "Computational Rules for Punning Riddles," *Int'l J. Humor Research*, vol. 10, no. 1, 1997, pp. 25–76.
4. O. Stock and C. Strapparava, "Getting Serious about the Development of Computational Humour," *Proc. 8th Int'l Joint Conf. Artificial Intelligence* (IJCAI 03), Morgan Kaufmann, 2003, pp. 59–64.
5. J. Taylor and L. Mazlack, "Computationally Recognizing Wordplay in Jokes," *Proc. 26th Ann. Conf. Cognitive Science Soc.* (CogSci 04), Lawrence Erlbaum Associates, 2004, pp. 1315–1320.
6. J. Wachman and R. Picard, "Tools for Browsing a TV Situation Comedy Based on Content Specific Attributes," *Multimedia Tools and Applications*, vol. 13, no. 3, 2001, pp. 255–284.

least one seed one-liner via a simple search using a Web search engine. Next, the algorithm parses the HTML Web pages, automatically identifies additional one-liners, and adds the one-liners to the seed set. The process is repeated until enough one-liners are collected.

An important aspect of any bootstrapping algorithm is the set of constraints used to steer the process and prevent the addition of noisy entries. Our algorithm uses a thematic constraint applied to Web page content and a structural constraint that exploits HTML annotations indicating similar text.

The algorithm implements the thematic constraint using a set of keywords, at least one of which must appear in a retrieved Web page's URL, potentially limiting the Web page's content to a theme related to that keyword. Our implementation's keyword set consisted of six words that explicitly indicate humor-related content: *oneliner*, *one-liner*, *humor*, *humour*, *joke*, and *funny*. For example, www.lohrere.com/joke004.html and www.mutedfaith.com/funny/life.htm satisfy this constraint.

The structural constraint exploits Web pages' HTML structure to identify enumerations of texts that include the seed one-liner. The basis for this is the hypothesis that because enumerations typically include texts of similar genre, a list including the seed one-liner would likely include additional one-line jokes. For instance, if a Web page has a seed one-liner preceded by the HTML tag <li> (that is, "list item"), other lines in the same enumeration that are preceded by the tag <li> are likely to be one-liners as well.

Two iterations of the bootstrapping process, started with a seed set of 10 one-liners, resulted in a set of about 24,000 one-liners. After removing the duplicates using a measure of string similarity based on the longest common subsequence metric, we had a final set of approximately 16,000 one-liners, which we used in the humor-recognition experiments.

Figure 2a illustrates the one-liner humor style. Because the collection process is automatic, noisy entries are possible. Manual verification of a random sample of 200 one-liners indicates an average of 9 percent potential noise in the data set, which is within reasonable limits as it does not appear to significantly impact the quality of the automatic learning process.

Automatic humor recognition

Once we had a large collection of humorous one-liners, we considered whether we could automatically recognize humor in text by modeling the humor recognition problem as a traditional machine learning task.

To test the hypothesis that automatic classification techniques are a viable approach to humor recognition,[4] we needed a data set containing both humorous (positive) and nonhumorous (negative) examples. Such data sets can be used to automatically learn computational models for humor recognition and at the same time evaluate such models' performance.

Learning is typically performed from both positive and negative examples. In addition to the humorous one-liners automatically collected from the Web, an automatic humor recognizer would therefore also require a collection of "negative" nonhumorous examples. We identified collections of sentences that were nonhumorous but had similar struc-

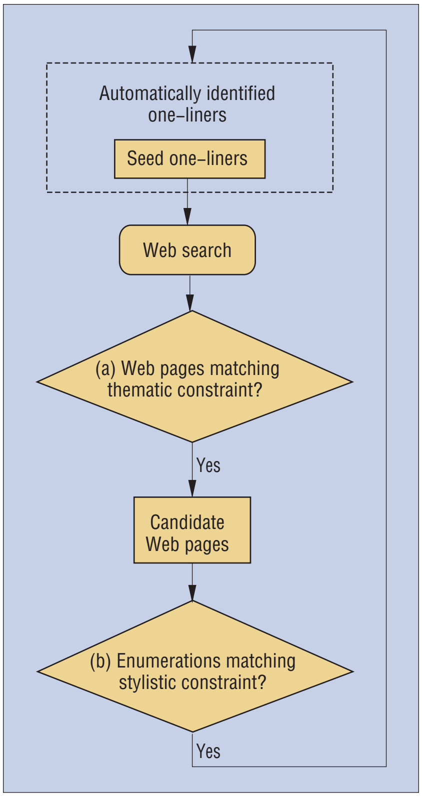

Automatically identified one–liners
Seed one–liners
Web search
(a) Web pages matching thematic constraint?
Yes
Candidate Web pages
(b) Enumerations matching stylistic constraint?
Yes

Figure 1. Web-based bootstrapping of one-liners using (a) thematic and (b) structural constraints.

One-liners
Take my advice; I don't use it anyway.
I get enough exercise just pushing my luck.
I just got lost in thought; it was unfamiliar territory.
Beauty is in the eye of the beer holder.
(a) I took an IQ test, and the results were negative.

Headlines
Trocadero expects tripling of revenues.
Silver fixes at two-month high, but gold lags.
Oil prices slip as refiners shop for bargains.
Japanese prime minister arrives in Mexico.
(b) Chains may raise prices after minimum wage hike.

Proverbs
Creativity is more important than knowledge.
Beauty is in the eye of the beholder.
Believe no tales from an enemy's tongue.
Do not look at the coat but at what is under the coat.
(c) A man is known by the company he keeps.

Figure 2. Examples of (a) one-liners, (b) headlines, and (c) proverbs.

ture and composition as the one-liners. We did not want the automatic classifiers to learn to distinguish between humorous and nonhumorous examples based simply on text length or obvious vocabulary differences. So, to help the classifiers identify humor-specific features, we chose similar negative and positive examples but with different comic effects.

The algorithm enforced structural similarity by requiring that the nonhumorous data set's examples all followed the same length restriction as the one-liners: one sentence with an average length of 10 to 15 words. We sought compositional similarity by identifying examples with similar creativity and intent as the one-liners.

We tested two sets of negative examples:

- *Headlines* from news articles published in the Reuters newswire from August 1996 to August 1997 (see figure 2b).[5] The headlines are short sentences with simple syntax and phrasing that tries to catch the reader's attention, similar to one-liners.
- *Proverbs* from an online proverb collection (see figure 2c). Proverbs are sayings that transmit, usually in a short sentence, important facts or experiences that many people consider true. They are condensed but memorable, similar to one-liners. In fact, some one-liners play on proverbs—for example, "Beauty is in the eye of the beer holder" instead of "Beauty is in the eye of the beholder."

To test the feasibility of automatically differentiating between humorous and nonhumorous texts using content-based features, our experiments formulated the humor-recognition task as a traditional text classification problem.

Briefly, a text classifier's goal is to learn the most representative features (typically words) associated with preclassified data so it can automatically classify new texts into a specified class (in this case, humorous or nonhumorous). Although numerous text categorization tools have proven successful, we chose two of the most frequently employed text classifiers—naive Bayes and support vector machines—on the basis of their performance in previous work and their diverse learning methodologies.

A *naive Bayes classifier* estimates a category's probability given a document using combined probabilities of words and categories. Naive Bayes classifiers assume word independence, but despite this simplification, they perform text classification well.[6] Of the two versions of naive Bayes classifiers (multinomial and multivariate Bernoulli), we use the multinomial model, which previous work has shown to be more effective.[7]

Support vector machines are binary classifiers that seek the hyperplane that best separates a set of positive examples from a set of negative examples with maximum margin.[8] Applications of SVM classifiers to text categorization led to some of the best results reported in the literature.[9]

The classification experiments were performed using stratified 10-fold cross validations for accurate evaluations. The baseline for all the experiments was 50 percent, which represents the classification accuracy obtained if a label of *humorous* (or *nonhumorous*) was assigned by default to all the data set's examples. Table 1 shows results we obtained using the two data sets and the naive Bayes and SVM classifiers. Figure 3 plots the learning curves.

The headlines' content appears to be most different from one-liners. Interestingly, content-based features can reliably distinguish proverbs from one-liners, indicating that despite their stylistic similarity, proverbs and one-liners deal with different topics.

Results from the automatic classification experiments reveal that computational approaches represent a viable solution for humor recognition and that good performance is possible using classification techniques based on stylistic and content features.

In addition to classification accuracy, we were also interested in classification performance variation with respect to data size, which is particularly relevant for directing future research. Depending on the learning curves' shape, we could decide to concen-

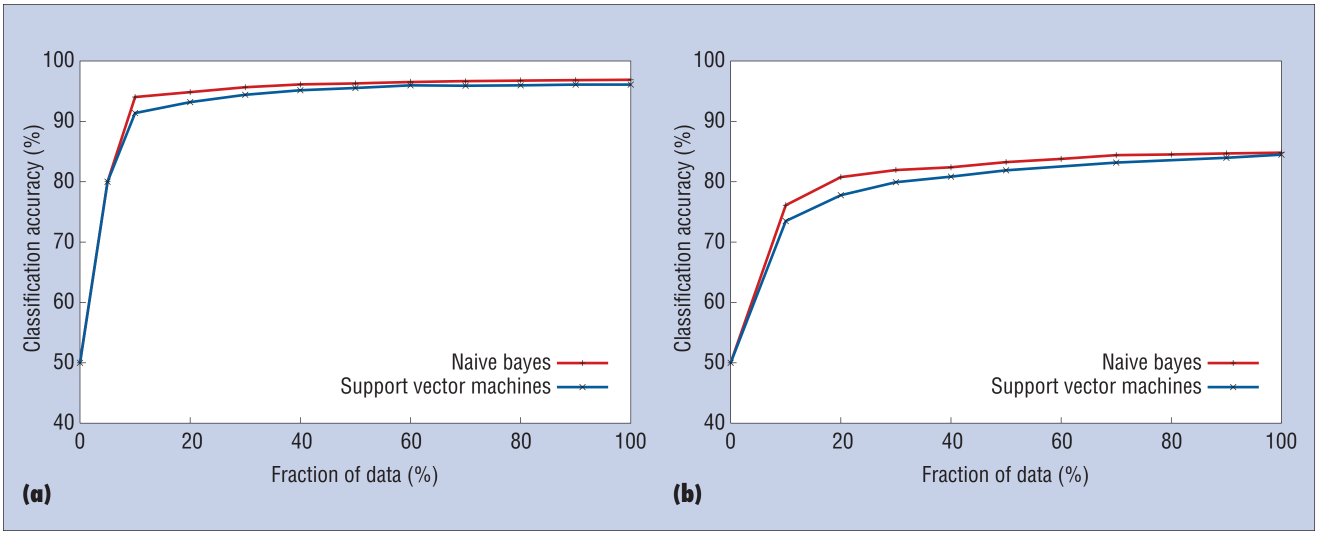


Figure 3. Learning curves for humor recognition using text classification techniques: (a) one-liners/headlines and (b) one-liners/proverbs.

trate future work either on acquiring larger data sets or identifying more sophisticated features. Figure 3 shows that regardless of the type of negative data or classification methodology, significant learning occurs only up to about 60 percent of the data (that is, about 10,000 positive examples and 10,000 negative examples). The curve's steep ascent, especially where learning begins, suggests that humorous and nonhumorous texts represent clearly distinguishable types of data. Toward the end of the learning, the curve for both classifiers becomes completely flat—probably because the data set contains noise, which starts to become visible for very large data sets. (We also like to think of this behavior as the computer losing its sense of humor after an overwhelming number of jokes, similar to how humans get bored and stop appreciating humor after hearing too many jokes). This plateau also suggests that more data are unlikely to improve an automatic humor-recognizer's quality, and more sophisticated features are probably necessary.

Adding humor to electronic applications

We also considered the possibility of automatically integrating humor into existing computer-based applications. Given that we can automatically construct very large collections of humorous texts, we wanted to use these collections to make humorous additions to existing computer-based applications. We believe that machines equipped with humorous capabilities will help induce users' emotions and beliefs and provide motivational support. Humor encourages creativity and changes user perspective. It can add value to many practical settings—for instance, humorous, attention-drawing items are key elements of product promotion and advertisement. Similarly, e-commerce applications could benefit from automatic and contextualized product promotion using humorous mottos and headlines, possibly tailored to user characteristics.

Entertainment and especially edutainment are perfect scenarios for computational humor, such as educational software that helps children explore word meanings while learning a second language. And another broad application area consists of communication tasks. Just as humans use humor to ease communication problems, human-computer interaction could use similar means to ease communication in applications such as dialogue systems, talking heads, embodied conversational agents, personal robots, and others.

Automatic insertion of humor into applications

A system would require two key language processing components to automatically add humor to existing text. First, we need a method to determine the most appropriate one-liner for a given context (such as email or lecture notes). This requires using a measure of semantic similarity between two input text segments. Second, we need a method to automatically assess a text's affective semantic orientation. Humor is not appropriate for all situations, so we want to determine whether a text's semantic orientation is happy or sad and inhibit the automatic addition of a one-liner when they are inappropriate.

Table 1. Humor-recognition accuracy using content-based features and naive Bayes and support vector machines classifiers.

Classifier	Headlines (%)	Proverbs (%)
Naive Bayes	96.67	84.81
Support vector machines	96.09	84.48

Text similarity. One of the most reliable measures of semantic similarity is latent semantic analysis.[10] In LSA, term co-occurrences in the documents of a corpus are captured by a dimensionality reduction operated by a singular value decomposition on the term-document matrix of the corpus. The SVD is similar to factor analysis, decomposing the matrix into three components. The first matrix contains vectors of orthogonal values from the original rows. The second matrix contains the k singular values of the original matrix. The third matrix contains vectors of orthogonal values from the original column values, such that by multiplying the three components, the original term-document matrix is recomposed.

LSA overcomes the problems (such as

sparseness and high dimensionality) of using a standard vector-space model. In fact, LSA computes similarity in a lower-dimensional space, which exploits second-order relations among terms and texts. LSA yields a vector-space model for homogeneously representing (and, hence, for comparing) words, sets of words, and texts. We apply LSA for text semantic similarity using the pseudo-document text representation for LSA computation. In practice, we represent each document (or set of words) in the LSA space by summing up the normalized LSA vectors of all the terms it contains, using a term frequency–inverse document frequency (*tf.idf*) weighting scheme. We then simply measure the resulting vector space's semantic similarity using the standard cosine similarity.

Text semantic orientation. Detecting the text semantic orientation is particularly important for avoiding inappropriate addition of humor to documents.

We detected a text's semantic orientation using a classifier trained on semantically labeled data, which automatically classifies a text as either happy or sad.[11] The corpus we used for training consists of a collection of blog posts from LiveJournal (www.livejournal.com), annotated with happy and sad moods. (Indicating a mood is optional, so the posts we use likely reflect the authors' true moods, because the authors chose to specify one.) The corpus consists of 10,000 blog posts, out of which half are happy and half are sad.

Using this corpus, we evaluated how well the two moods could be separated by their linguistic content, a possible indication of the mood annotations' reliability. We trained a naive Bayes text classifier over unigram features and evaluated the classification accuracy in a five-fold cross-validation experiment. We measured the accuracy at 79.13 percent, a significant improvement over the 50 percent accuracy associated with the naive baseline of using one mood assignment by default. This data set's annotations are clearly separable, and therefore we can use this corpus to learn the characteristics of the happy and sad moods—and, consequently, to automatically label a new text's semantic orientation.

Applications. We tested our method for automatically inserting humor in two applications—Fun Email, which adds a one-liner at the end of an email, and Fun Lecture, which adds one-liners to lecture material.

Application 1: Fun Email

We modified SquirrelMail, a popular open source mail client, when creating Fun Email, but other mail client choices are also feasible. The Fun Email system operates at the mail-client level, with the user selecting from two basic modalities:

- annotating every email that the user receives with a one-liner, automatically choosing a one-liner appropriate to the email's context and adding it to the end of the message; or
- annotating every email that the user sends with a one-liner, which functions as a dynamic and contextual signature slogan.

Overall, the user study shows that humorous additions to emails can significantly improve the user experience in an email application, especially if the humorous texts are appropriate.

The system's architecture is simple and includes the two main language processing components we mentioned earlier. The SVD operation is performed offline, starting with the corpus of 16,000 one-liners, after removing stop words and considering only the first 100 dimensions. The lexicon in the LSA space consists of about 18,000 words.

For each candidate email, Fun Email

- determines if the content type is text/plain, because the system can only process text files;
- finds the text's semantic orientation and ignores the email if adding humor would be inappropriate;
- extracts the last 30 percent of text from the email body so that the similarity is computed with respect to the topic of the last part of the email; and
- compares the body's LSA vector with those of the one-liners, identifying the most similar (and most appropriate).

Figure 4 shows an email with an automatically inserted one-liner.

We conducted a user study using a set of 10 emails covering different topics, each having three different versions: a basic version that contains just the email's text, a random version that selects a motto randomly from the one-liner data set and adds it at the end of the email, and a Fun Email version, which adds a closing one-liner using the Fun Email system.

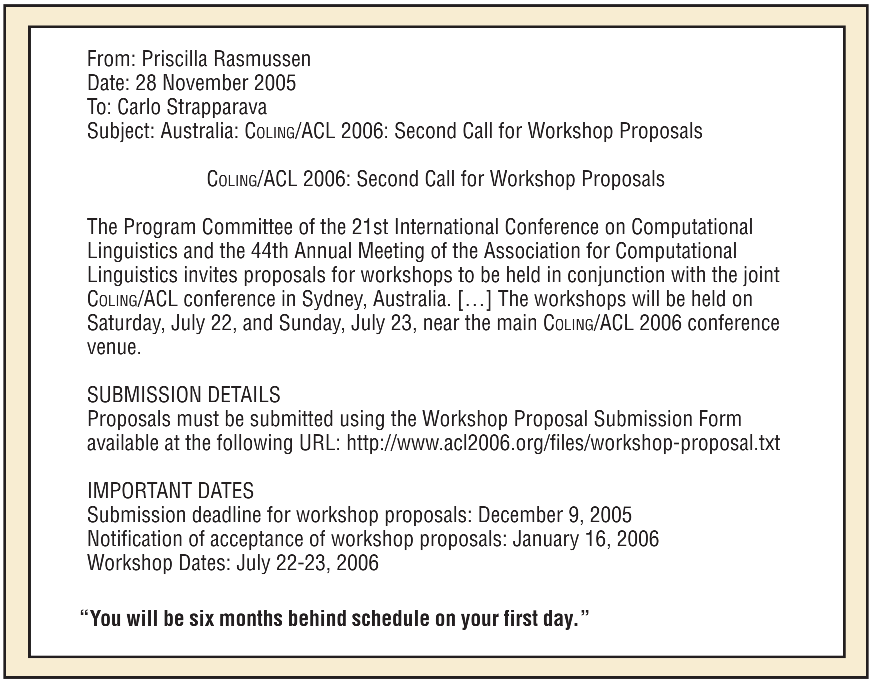
A group of 13 users participated in the evaluation. Users had different professions (students, professors, and administration staff) and different cultural backgrounds (American, Chinese, Hungarian, Italian, and Romanian). We asked the users to score each email version on a 10-point scale in four dimensions—entertainment (the email was entertaining), appropriateness (the motto was appropriate), intelligence (the email program behaved intelligently), and adoption (I would use the email program myself).

Figure 5 shows the study's results. Perhaps not surprisingly, users clearly preferred the Fun Email output to the other two alternatives and showed high interest in adopting it. The random one-liner selection was also moderately successful, although in most cases, the users did not think the humorous additions suited the content of the email. Overall, the user study shows that humorous additions to emails can significantly improve the user experience in an email application, especially if the humorous texts are appropriate.

Application 2: Fun Lecture

Humor can improve the quality and efficiency of teaching and make it more enjoyable. Previous research found that using humor in lectures improved the students' attention and helped them learn with greater interest and joy.[12,13] Humor also stimulated creativity and improved students' creative participation in class. Finally, during tests, humor helped students lower their anxiety levels and improve their performance.

Educational environments and intelligent tutoring systems can include computational models of humor as a teaching style or model. In particular, instructors can use computational humor during lesson preparation—perhaps by adding one-liners to teaching materials. For example, the following one-liner can improve student attention and understanding of binary encoding in a computer class: "There are 10 types of people in this world; those who understand binary and those who don't."



From: Priscilla Rasmussen
Date: 28 November 2005
To: Carlo Strapparava
Subject: Australia: COLING/ACL 2006: Second Call for Workshop Proposals

COLING/ACL 2006: Second Call for Workshop Proposals

The Program Committee of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics invites proposals for workshops to be held in conjunction with the joint COLING/ACL conference in Sydney, Australia. […] The workshops will be held on Saturday, July 22, and Sunday, July 23, near the main COLING/ACL 2006 conference venue.

SUBMISSION DETAILS
Proposals must be submitted using the Workshop Proposal Submission Form available at the following URL: http://www.acl2006.org/files/workshop-proposal.txt

IMPORTANT DATES
Submission deadline for workshop proposals: December 9, 2005
Notification of acceptance of workshop proposals: January 16, 2006
Workshop Dates: July 22-23, 2006

"You will be six months behind schedule on your first day."

Figure 4. An email with an automatically inserted one-liner.

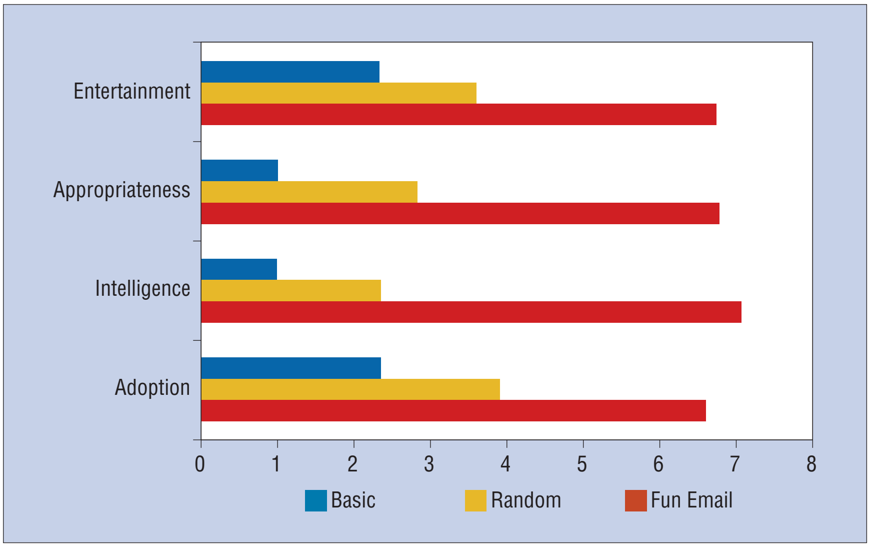


Figure 5. User study results from the Fun Email evaluation.

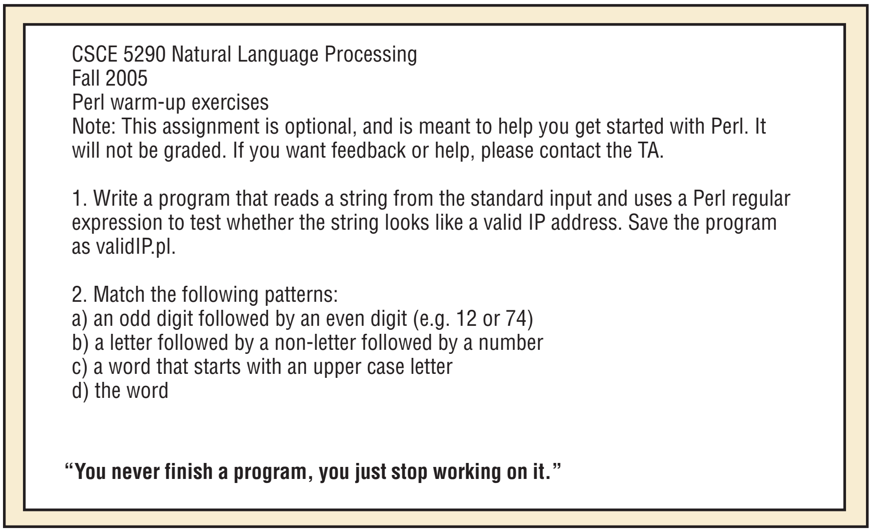
As in the Fun Email application, humorous additions must be made according to the context. The core mechanism of adding humor is similar to that of Fun Email. Provided with a document containing lecture notes or an assignment, the system will determine the most appropriate one-liner for the document's context. It will not assign more than two or three one-liners to a lecture document. Also, regardless of an assignment's length, Fun Lecture will add just one one-liner. Figure 6 shows an output of Fun Lecture on an assignment.

We are still evaluating Fun Lecture. We are currently using the system to automatically make humorous additions to the lecture notes and assignments of a class that Rada Mihalcea is teaching, and we are collecting feedback from the students. In a preliminary evaluation, we augmented a set of preparatory exam questions with a one-liner and handed it out to 24 students. The evaluation (also on a 10-point scale) indicated that the students enjoyed the humorous addition (6.0) and wanted to see more humorous content in lecture materials (6.5). Moreover, although it is difficult to quantify, the atmosphere during the trial was more relaxed than usual—most likely triggered by the laughs at the beginning of class when the students discovered the one-liners. Indeed, although it is easy to carry out an evaluation of the one-liner's appropriateness and pleasantness, it is more difficult to organize an evaluation on the one-liner's effect on the student learning process.

In the future, we plan to investigate whether we can apply our bootstrapping technique equally well to other creative genres besides one-liners. We will also refine our Fun Email and Fun Lecture applications and investigate other scenarios that could benefit from the addition of humor and the creative use of natural language. ■

References

1. O. Stock and C. Strapparava, "Getting Serious about the Development of Computational Humour," *Proc. 8th Int'l Joint Conf. Artificial Intelligence* (IJCAI 03), Morgan Kaufmann, 2003, pp. 59–64.
2. K. Binsted and G. Ritchie, "Computational Rules for Punning Riddles," *Int'l J. Humor Research*, vol. 10, no. 1, 1997, pp. 25–76.
3. J. Taylor and L. Mazlack, "Computationally Recognizing Wordplay in Jokes," *Proc. 26th Ann. Conf. Cognitive Science Soc.* (CogSci 04), Lawrence Erlbaum Associates, 2004, pp. 1315–1320.
4. R. Mihalcea and C. Strapparava, "Computational Laughing: Automatic Recognition of Humorous One-liners," *Proc. 27th Ann. Conf. Cognitive Science Soc.* (CogSci 05), Lawrence Erlbaum Associates, 2005, pp. 1513–1518; www.cogsci.rpi.edu/CSJarchive/Proceedings/2005/docs/p1513.pdf.
5. D. Lewis et al., "RCV1: A New Benchmark Collection for Text Categorization Research," *J. Machine Learning Research*, vol. 5, 2004, pp. 361–397.
6. Y. Yang and X. Liu, "A Re-examination of Text Categorization Methods," *Proc. 22nd Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval*, ACM Press, 1999, pp. 42–49.
7. A. McCallum and K. Nigam, "A Comparison of Event Models for Naive Bayes Text Classification," *Proc. AAAI-98 Workshop on*



CSCE 5290 Natural Language Processing
Fall 2005
Perl warm-up exercises
Note: This assignment is optional, and is meant to help you get started with Perl. It will not be graded. If you want feedback or help, please contact the TA.

1. Write a program that reads a string from the standard input and uses a Perl regular expression to test whether the string looks like a valid IP address. Save the program as validIP.pl.

2. Match the following patterns:
a) an odd digit followed by an even digit (e.g. 12 or 74)
b) a letter followed by a non-letter followed by a number
c) a word that starts with an upper case letter
d) the word

"You never finish a program, you just stop working on it."

Figure 6. An assignment with an automatically inserted one-liner.

Learning for Text Categorization, AAAI Press, 1998, pp. 41-49.

8. V. Vapnik, *The Nature of Statistical Learning Theory*, Springer, 1995.

9. T. Joachims, "Text Categorization with Support Vector Machines: Learning with Many Relevant Features," *Proc. 10th European Conf. Machine Learning* (ECML 98), Springer, 1998, pp. 137–142.

10. T.K. Landauer, P. Foltz, and D. Laham, "Introduction to Latent Semantic Analysis," *Discourse Processes*, vol. 25, 1998, pp. 259–284.

11. R. Mihalcea and H. Liu, "A Corpus-Based Approach to Finding Happiness," *Proc. AAAI Spring 2006 Symp. Computational Approaches to Analyzing Weblogs*, AAAI Press, 2006 pp. 139–144.

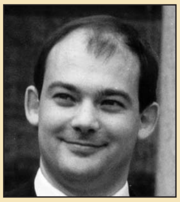
12. P. McGhee, *Humor and Children's Development: A Guide to Practical Applications*, Haworth Press, 1989.

13. J. Bryant and D. Zillman, "Using Humor to Promote Learning in the Classroom," *Humor and Children's Development: A Guide to Practical Applications,* Paul E. McGhee, ed., Haworth Press, 1989, pp. 49–78.

For more information on this or any other computing topic, please visit our Digital Library at www.computer.org/publications/dlib.

The Authors

Rada Mihalcea is an assistant professor of computer science at the University of North Texas. Her research interests include word sense disambiguation, text summarization and keyphrase extraction for very large documents, (nontraditional) methods for building annotated corpora with volunteer contributions over the Web, graph-based algorithms for text processing, and computational humor. She received her PhD in computer science from Southern Methodist University. She's the president of the Association for Computational Linguistics Special Group on the Lexicon (SIGLEX) and a board member for the ACL Special Group on Natural Language Learning (SIGNLL). Contact her at the Dept. of Computer Science, Univ. of North Texas, PO Box 311366, Denton, TX, 76203; rada@cs.unt.edu; www.cs.unt.edu/~rada.



Carlo Strapparava is a senior researcher at ITC-irst (Istituto per la ricerca scientifica e Tecnologica) in the Communication and Cognitive Technologies Division. His research activity covers AI, natural language processing, intelligent interfaces, cognitive science, knowledge-based systems, user models, adaptive hypermedia, lexical knowledge bases, word sense disambiguation, and computational humor. He received his doctorate in computer science from the University of Pisa, Italy. Contact him at ITC-irst, via Sommarive 18, Povo, I-38050 Trento, Italy; strappa@itc.it; http://tcc.itc.it/people/strapparava.html.