Understanding Eggcorns

Sravana Reddy
Department of Computer Science
The University of Chicago
sravana@cs.uchicago.edu

Abstract

An eggcorn is a type of linguistic error where a word is substituted with one that is *semantically plausible* – that is, the substitution is a semantic reanalysis of what may be a rare, archaic, or otherwise opaque term. We build a system that, given the original word and its eggcorn form, finds a semantic path between the two. Based on these paths, we derive a typology that reflects the different classes of semantic reinterpretation underlying eggcorns.

1 Introduction

The term "eggcorn" was coined in 2003 by Geoffrey Pullum (Liberman, 2003) to refer to a certain type of linguistic error where a word or phrase is replaced with one that is phonetically similar *and* semantically justifiable. The eponymous example is *acorn* → *eggcorn*, the meaning of the latter form being derived from the acorn's *egg*-like shape and the fact that it is a seed (giving rise to *corn*). These errors are distinct from mere misspellings or mispronunciations in that the changed form is an alternate interpretation of the original.

The reinterpretation may be related to either the word's perceived meaning or etymology (as in the case of *acorn*), or some context in which the word is commonly used. In this sense, eggcorns are similar to *folk etymologies* – errors arising from the misinterpretation of borrowed or archaic words – with the difference being that the latter are adopted by an entire culture or linguistic community, while eggcorns are errors made by one or more individual speakers.

The formation of eggcorns and folk etymologies, mistakes though they are, involves a creative leap within phonetic and semantic constraints (much like what is required for puns or certain classes of jokes). Eggcorns range from simple reshapings of foreign words (*paprika* → *pepperika*) and substitutions from similar domains (*marshal* → *martial*), to the subtly clever (*integrate* → *intergrade*), the technological (*sound bite* → *sound byte*), or the funny (*stark-raving mad* → *star-craving mad*). The source of reinterpretation may be a weak imagined link (*wind turbine* → *wind turban*), or an invented myth (*give up the ghost* → *give up the goat*[1]). And often, it is not clear what the exact link is between the derived and the original forms, although it is usually obvious (to the human eye) that there is a connection.

This paper explores some ways of **automatically tracing the link** between a word and its eggcorn.

In reality, we are chiefly concerned with computing the connections between a *word and its reinterpreted form*. Such pairs may also occur as folk etymologies, puns, riddles, or get used as a poetic device. However, we use eggcorns as a testbed for three main reasons: there are a number of documented examples, the reanalyses are *accidental* (meaning the semantic links are more unpredictable and tenuous than in the cases of deliberate reshapings), and the errors are idiosyncratic and relatively modern – and hence have not been fossilized in the lexicon – making them transparent to analysis (as opposed to many folk etymologies and other historical errors). That said, much of the work described here can be potentially applied to other instances of semantic reinterpretation as well.

[1] http://eggcorns.lascribe.net/english/714/goat/

Proceedings of the NAACL HLT Workshop on Computational Approaches to Linguistic Creativity, pages 17–23,
Boulder, Colorado, June 2009.

The first part of the paper describes an algorithm (the "Cornalyzer") for finding a semantic path between the original and reinterpreted forms of an eggcorn pair. We then proceed to use the results of this algorithm to *cluster* the eggcorn examples into 5 classes, with a view to learning a typology of eggcorns.

2 Related Work

One work related to this area (Nelken and Yamangil, 2008) uses Wikipedia to automatically mine eggcorns by searching for pairs of phonemically similar words that occur in the same sentence context in different revisions. However, the mined examples are reported to contain many false positives since the algorithm does not include a notion of semantic similarity.

Folk etymologies, the closest cousin to eggcorns, have been studied from a linguistic point of view, including some of the same questions we tackle here (only, not from a computational side) – how is an new word derived from the original, and what are the different categories of folk etymologies? (Rundblad and Kronenfeld, 1998), (Scholfield, 1988). To the best of our knowledge, there has been no previous work in inducing or computationally understanding properties of neologisms and errors derived through misinterpretation. However, there is a substantial literature on algorithmic humor, some of which uses semantic relationships – (Stock and Strapparava, 2006), (Manurung et al., 2008), among others.

3 Data

The list of eggcorns is taken directly from the **Eggcorn Database**[2] as of the submission date. To assure soundness of the data, we include only those examples whose usage is attested and which are confirmed to be valid and contemporary reanalyses[3], giving a total of 509 instances. Table 1 shows a sample of the data.

Every example can be denoted by the tuple (w, e, c) where c is the list of obligatory *contexts* in which the reanalysis takes place, w is the original form, and e is the modified (eggcorned) form.

Table 1: A few eggcorns. 'X' can be replaced for w or e to give the original form in context, or the eggcorn in context respectively.

Original form w	Changed form e	Context c
bludgeon	bloodgeon	X
few	view	name a X
entree	ontray	X
praying	preying	X mantis
jaw	jar	X-dropping
dissonance	dissidence	cognitive X

The Cornalyzer uses WordNet (Fellbaum, 1998) version 3.0, including the built-in morphological tools for lemmatization and dictionary definitions[4].

4 Automated Understanding of Eggcorn Generation

Broadly speaking, there are two types of eggcorns:

1. Ones where e or a part of e is semantically related to the original word w (*lost* → *loss* in 'no love lost') or the context c (*pied* → *pipe* in 'pied-piper').

2. Eggcorns where e is related to an image or object that is *connected to* or evoked by the original (like 'song' in *lip-sync* → *lip-sing*).

For the first, a database of semantic relations between words (like WordNet) can be used to find a semantic connection between w and e. The second type is more difficult since external knowledge is needed to make the connection. To this end, we make use of the "glosses" – dictionary definitions of word senses – included in WordNet. For instance, the 'lip-sing' eggcorn is difficult to analyze using *only* semantic relations, since neither 'sync' nor 'lip' are connected closely to the word 'sing'. However, the presence of the word *song* in the gloss of *lip-sync*:

```
move the lips in synchronization
(with recorded speech or song)
```

[2] http://eggcorns.lascribe.net/

[3] In other words, all examples that are classified as 'questionable' (or otherwise indicated as being questionable), 'not an eggcorn', 'citational' or 'nearly-mainstream' are eliminated.

[4] From http://wordnet.princeton.edu/

makes the semantic connection fairly transparent.

The Cornalyzer first attempts to analyze an eggcorn tuple (w, e, c) using semantic relations (§4.1). If no sufficiently short semantic path is found, the eggcorn is presumed to be of the second type, and is analyzed using a combination of semantic relations and dictionary glosses (§4.2).

4.1 Analysis using Word Relations

4.1.1 Building the Semantic Graph

WordNet is a semantic dictionary of English, containing a list of *synsets*. Each synset consists of a set of synonymous words or collocations, and its relations (like hypernymy, antonymy, or meronymy) with other synsets. The dictionary also includes *lexical relations* – relations between words rather than synsets (for instance, a *pertainym* of a noun is an adjective that is derived from the noun).

WordNet relations have been used to quantify semantic similarity between words for a variety of applications (see Budanitsky and Hirst (2001) for a review of similarity measures). The Cornalyzer uses the same basic idea as most existing measures – finding the shortest path between the two words – with some modifications to fit our problem.

We adopt the convention that two words w_1 and w_2 have the relation R if they are in different synsets S_1 and S_2, and $R(S_1, S_2)$ is true. We also define two new lexical relations that are not directly indicated in the dictionary: w_1 and w_2 are *synonyms* if they are in the same synset, and *homographs* if they have identical orthographic forms and lexical categories but are in different synsets. [5]

This relational network can hence be used to define a graph G_s over words, where there is an edge of type t_R from w_1 to w_2 if $R(w_1, w_2)$ holds. Some of the relations in WordNet (like antonymy) are ignored, either because they invert semantic similarity, or are not sufficiently informative. Table 2 summarizes the relations used.

This graph can be used to find the semantic relationships between an original word w and its eggcorn form e, if both forms are in the dictionary, and there exists a path from w to e. However, it is often the case that e or w are not in the dictionary, or that a path does not exist. This could be because one of the forms is an inflected form or compound, or that some substring of e – rather than the whole word or collocation – is the reinterpreted segment. It is also essential to consider the strings in c, since many eggcorns result from semantic reinterpretation of the contexts.

Table 2: WordNet relations used to build the semantic graph.

Relation	Parts of Speech	Reflexive Relation	Example
Synonym	(N, N) (V, V) (Adj, Adj) (Adv, Adv)	Synonym	(forest, wood) (move, displace) (direct, lineal) (directly, at once)
Homograph	(All, All)	Homograph	(call [greet], call [order])
Hypernym	(V, V) (N, N)	Troponym/ Hyponym	(move, jump) (canine, fox)
Meronym	(N, N)	Holonym	(forest, tree)
Has Instance	(N, N)	Instance Of	(city, Dresden)
Cause	(V, V)	Caused by	(affect, feel)
Entails	(V, V)	not specified	(watch, look)
Similar To	(Adj, Adj)	Similar To	(lucid, clear)
Related	(V, V) (Adj, Ad)	Related	(few, some)
Same Group	(V, V)	Same Group	(displace, travel)
Has Attribute	(Adj, N)	Attribute Of	(few, numerousness)
Derivational Relation	(N, V) (N, Adj) (V, Adj)	Derivational Relation	(movement, move) (movement, motional) (move, movable)
Pertainym	(Adj, N) (Adv, Adj)	not specified	(direct, directness) (directly, direct)

Hence, three new non-semantic relations are defined: w_1 is a *substring* of w_2 if the orthographic form of w_1 is a substring of that of w_2, and w_1 and w_2 are *contextually linked* if they occur in the same collocation or compound. If w_2 can be derived from w_1 using WordNet's lemmatizer, w_2 is an *inflected* form of w_1.

A new graph G_e is constructed by adding edges of types $t_{substring}$, $t_{context}$, and $t_{inflect}$ to G_s. For all eggcorn tuples (w, e, c):

1. If e or w are not in the dictionary, add them to G_e as a vertex
2. Add edges of type *inflect* between e and its base form.
3. Add edges of type *substring* from e to every

[5] This paper uses 'word' to include sense – i.e, 'bank' as in *slope beside a body of water* and 'bank' as in *financial institution* are distinct. When required for disambiguation, the WordNet *sense number*, which is the index of the sense in the list of the word's senses, is added in parenthesis; e.g. bank (2) for the *financial institution* sense.

substring of length ≥ 3 that is in the dictionary (except those substrings which are base forms of e), and edges of type *supstring* in the other direction.

4. Extract a set of 'context words' from c by splitting it along spaces and hyphenation. Select those words which are in the dictionary.

5. Add edges of type *context* from w and e to each extracted context word.

For example, given the data in table 1, the following vertices and edges will be added to G_e:

Vertices bloodgeon, ontray, preying, praying

Substring edges (bloodgeon, blood), (bloodgeon, loo), (bloodgeon, eon), (view, vie), (entree, tree), (ontray, ray), (ontray, tray)

Superstring edges above edges in the other direction

Inflectional edges (preying, prey), (praying, pray). These edges are bidirectional.

Context edges (few, name), (view, name), (few, a), (view, a), (praying, mantis), (preying, mantis), (jaw, dropping), (jar, dropping), (dissonance, cognitive), (dissidence, cognitive). These edges are also bidirectional.

4.1.2 Tracing the Semantic Path

Given the semantic graph, our working assumption is that e is generated from w by following the shortest path from w to e (denoted by $P(w, e, c)$).

1. If w and e are both in the dictionary, find $P_1(w, e) =$ the shortest path from w to e in G_s

2. Find $P_2(w, e, c) =$ the shortest path using substrings of e and/or c in G_e

(Since the edges are unweighted, the shortest path from w to e is found simply by performing breadth-first search starting at w.)

$P(w, e, c)$ is simply the shorter of $P_1(w, e)$ (if it exists) and $P_2(w, e, c)$. Note that there may be several shortest paths, especially since words that are synonymous have almost the same incident semantic edges. Since the candidate shortest paths generally do not differ much from one another (as far as their semantic implications), an arbitrary path is chosen to be P.

Table 3 shows the paths found by the algorithm for some eggcorns.

4.2 Analysis using Dictionary Definitions

As described in §4, the source of many eggcorns is knowledge external to the original word or contexts through some concept or object suggested by the original. In such cases, a semantic network will not suffice to find the reinterpretation path. One possible way of accessing the additional information is to search for w and e in a large corpus, and extract the key words that appear in conjunction with these forms.

However, filtering and extracting the representative information can quickly become a complex problem beyond the scope of this paper. Hence, as a first approximation, we use the dictionary definitions (glosses) that accompany synsets in WordNet. To optimize efficiency and to avoid having noise added by the definitions, the Cornalyzer only resorts to this step if a *sufficiently short path* – that is, a path of length $\leq k$ for some threshold k – is not found when only using word relations. (The results suggest 7 as a good threshold, since most of discovered paths that are longer than 7 tend not to reflect the semantic relationships between the eggcorn and the original form.)

Every gloss from all senses of a lexical item[6] x (for all x in the dictionary) is first tokenized, and punctuation stripped. All tokens are stemmed using the built-in lemmatizer. Only those tokens t that are already present as vertices in G_e are taken into consideration. However, it should be clear that not all tokens t are equally relevant to x. For instance, consider one gloss of the noun "move":

> the act of changing location from one place to another

which gives the tokens *act*, *changing*, *location*, *one*, *place*, *another*. Clearly, the tokens *changing*, *location*, and *place* rank higher than the others in terms of how indicative they are of the meaning of the noun.

[6] A lexical item is a word independent of sense, e.g, all senses of 'bank' constitute a single lexical item.

Table 3: A sample of semantic similarity paths. $x \xrightarrow{R} y$ means "y is an R of x". When relevant, WordNet sense numbers are indicated.

Eggcorn tuple $(word, eggcorn, context)$	Path from word to eggcorn
(mince, mix, 'X words')	mince $\xrightarrow{hypernym}$ change $\xrightarrow{hyponym}$ mix
(few, view, 'name a X')	few $\xrightarrow{deriv}$ fewness $\xrightarrow{hypernym}$ number $\xrightarrow{hypernym}$ amount $\xrightarrow{hypernym}$ magnitude $\xrightarrow{hyponym}$ extent $\xrightarrow{hyponym}$ scope $\xrightarrow{hyponym}$ view
(dissonance, dissidence, cognitive X)	dissonance $\xrightarrow{synonym}$ disagreement (1) $\xrightarrow{homograph}$ disagreement (3) $\xrightarrow{hyponym}$ dissidence
(ado, [to-do, to do], ['much X about nothing', 'without further X'])	ado $\xrightarrow{synonym}$ stir (3) $\xrightarrow{homograph}$ stir (1) $\xrightarrow{hypernym}$ to-do
(jaw, jar, X-dropping)	jaw $\xrightarrow{context}$ dropping $\xrightarrow{inflect}$ drop $\xrightarrow{hypernym}$ displace $\xrightarrow{hyponym}$ jar
(ruckus, raucous, X)	ruckus $\xrightarrow{homograph}$ din $\xrightarrow{deriv}$ cacophonous $\xrightarrow{similar}$ raucous
(segue, segway, X)	segue $\xrightarrow{hypernym}$ passage (1) $\xrightarrow{homograph}$ passage (3) $\xrightarrow{hypernym}$ way $\xrightarrow{supstring}$ segway

One way of reflecting these distinctions in the Cornalyzer is to *weight* these terms appropriately, with something resembling the TF-IDF (Salton and Buckley, 1988) measure used in information retrieval. Let $tf(t,x)$ = the frequency of the token t in the glosses of x, and $idf(t) = \log \frac{N}{df(t)}$ where N = the number of lexical items in the dictionary and $df(t)$ = the number of lexical items in the dictionary whose glosses contain t. Define $W(t,x) = tf(t,x) \cdot idf(t)$.

A new graph G_d is constructed from G_e by adding edges of type *hasdef* from every lexical item x to tokens t in its glosses with the edge-weight $1 + 1/W(t,x)$, and reflexive edges of type *indef* from t to x with the same weight. All existing edges in the original graph G_e are assigned the weight 1.

The semantic path from w to e is found by the process similar to what was described in §4.1.2: first find $P_1(w,e)$ and $P_2(w,e,c)$ as well as $P_3(w,e,c)$ = the shortest path from w to e in G_d, and let $P(w,e,c)$ be the shortest of the three. Since G_d has weighted edges, the shortest path P_3 is computed using Dijkstra's algorithm.

Dictionary-definition-based paths P_2 for some eggcorns are shown in Table 4. The shortest P_2 paths are also shown for comparison. The P_3 paths generally appear to be closer to a human judgment of what the semantic reinterpretation constitutes. In the case of *(bludgeon → bloodgeon)*, for example, P_2 shows no indication of the key connection (bleeding due to being bludgeoned), whereas P_3 captures it perfectly.

Of the 509 eggcorns, paths were found for 238 instances by using only G_s or G_e as the relational graph. Paths for a total of 372 eggcorns were found when using dictionary glosses in the graph G_d.

5 From Generation to Typology

A quick glance at tables 3 and 4 shows that the paths vary in shape and structure: some paths move up and down the hypernym/homonym tree, while others move laterally along synonyms and polysemes; some use no external knowledge, while others make primary use of context information and dictionary glosses. A natural next step, therefore, is to *group the eggcorns* into some number of classes that represent general categories of semantic reanalysis. We can achieve this by clustering eggcorns based on their semantic shortest paths.

5.1 Clustering of Paths

One natural choice for a feature space is the set of all 24 relations (edge-types) used in G_d. An eggcorn (w,e,c) is represented as a vector $[v_1, v_2, \ldots v_{24}]$ where v_i = the number of times that relation R_i (or the reflexive relation of R_i) appears in $P(w,e,c)$.

These vectors are then clustered using k-means

Table 4: Some semantic paths using dictionary glosses. As before, $x \xrightarrow{R} y$ stands for "y is an R of x", and the numbers in parentheses following a lexical item are the WordNet sense numbers corresponding to that word.

Eggcorn tuple	Path from word to eggcorn
(bludgeon, bloodgeon, X)	P_3 (length 6): bludgeon $\xrightarrow{hypernym}$ hit (3) $\xrightarrow{homograph}$ hit (6) $\xrightarrow{hypernym}$ wound $\xrightarrow{indef}$ gore $\xrightarrow{hypernym}$ blood $\xrightarrow{supstring}$ bloodgeon
	P_2 (length 11): bludgeon $\xrightarrow{hypernym}$ club $\xrightarrow{hypernym}$ stick $\xrightarrow{hypernym}$ implement $\xrightarrow{hypernym}$ instrumentality $\xrightarrow{hypernym}$ artefact $\xrightarrow{hyponym}$ structure $\xrightarrow{hyponym}$ area $\xrightarrow{hyponym}$ room $\xrightarrow{hyponym}$ lavatory $\xrightarrow{hyponym}$ loo $\xrightarrow{supstring}$ bloodgeon
(entree, [ontray, on-tray], X)	P_3 (length 4): entree $\xrightarrow{indef}$ meal $\xrightarrow{indef}$ food $\xrightarrow{hasdef}$ tray $\xrightarrow{supstring}$ ontray
	P_2 (length 8): entree $\xrightarrow{hyponym}$ plate (8) $\xrightarrow{homograph}$ plate (4) $\xrightarrow{hypernym}$ flatware $\xrightarrow{hypernym}$ tableware $\xrightarrow{hyponym}$ tea set $\xrightarrow{meronym}$ tea tray $\xrightarrow{hypernym}$ tray $\xrightarrow{supstring}$ on-tray
(praying, preying, X mantis)	P_3 (length 6): praying $\xrightarrow{context}$ mantis $\xrightarrow{indef}$ predacious $\xrightarrow{synonym}$ predatory (3) $\xrightarrow{homograph}$ predatory (2) $\xrightarrow{indef}$ prey $\xrightarrow{inflect}$ preying
	P_2 (length 8): praying $\xrightarrow{context}$ mantis $\xrightarrow{hypernym}$ dictyopterous insect $\xrightarrow{hypernym}$ insect $\xrightarrow{hypernym}$ arthropod $\xrightarrow{hypernym}$ invertebrate $\xrightarrow{hypernym}$ animal $\xrightarrow{hyponym}$ prey $\xrightarrow{inflect}$ preying

and a Euclidean distance metric. We experimented with a few different values of k and found that $k = 5$ produces clusters that are the most semantically coherent.

5.2 Results

The five clusters roughly correspond to the each of the following characteristic paths $P(w, e, c)$:

1. Independent of dictionary glosses and of context, and mostly contain *synonym*, *homograph*, *related*, or *similar to* types of edges.
2. Contain several *hypernym* and *hyponym* edges.
3. Contain several *substring*, *supstring*, and *inflect* or *derivational* edges.
4. Heavily dependent on *context* edges.
5. Heavily dependent on dictionary glosses.

Eggcorns in these clusters can be interpreted to be (1) **Near-synonyms**, (2) **Semantic cousins** – deriving from a common general concept or entity, (3) **Segmentally related** – being linked by morphological operations, (4) **Contextually similar**, or (5) **Linked by implication** – deriving from an implicit concept.

A sample of the cluster membership is shown in Table 5.

6 Discussion

This paper presents a procedure for computationally understanding the semantic reanalyses of words. We identified the two general types of eggcorns, and built the appropriate networks overlying the WordNet graph and dictionary in order to trace the semantic path from a word to its eggcorn.

An obvious drawback to our method stems from the fact that the semantic dictionary is not perfect, or fully reflective of human information. Similarly, dictionary glosses are a limited source of external information. It would hence be worth exploring data-driven methods to augment a source like WordNet, such as building a word graph from co-occurrences in text, or using corpora to derive distributional similarity measures.

The Cornalyzer is only an exploratory first step – there are a wealth of other possible computational problems related to eggcorns. Semantic path-finding can be extended to defining some measure of eggcorn strength or plausibility. The algorithm can also be used to mine for new eggcorns – a threshold or a set of criteria for an 'eggcornish' path can

Table 5: A look at the clustered eggcorns.

Cluster	Examples
1	(cognitive dissonance → cognitive dissidence), (ado → to-do), (slake thirst → slack thirst), (ruckus → raucous), (sparkle (protests, etc) → spark), (poise to do → pose to do), ...
2	(sow wild oats → sow wild oaks), (name a few → name a view), (whet, wet), (curb hunger → curve hunger), (entree → ontray), (mince words → mix words), ...
3	(utmost → upmost), (valedictorian → valevictorian), (quote unquote → quote on quote), (playwright → playwrite), (no love lost → no love loss), (snub → snob), ...
4	(pied piper → pipe piper), (powerhouse → powerhorse), (jaw-dropping → jar-dropping), (sell (something) down the river → sail (something) down the river), ...
5	(renowned, reknowned), (praying mantis → preying mantis), (expatriate → expatriot), (skim milk → skimp milk), (sopping wet → soaping wet), (pique → peak), ...

be set based on the paths found for known eggcorns, thus helping separate them from false positives (typos and misspellings).

Another possible line of work is finding generalizations in *pronunciation* changes from the original. "The Eggcorn Database" website includes a partial catalogue of phonetic changes like *t-flapping* and *cot/caught merger* – it would be interesting to see if such patterns and categories can be learnt. The basic model of the Cornalyzer can potentially also be extended to applications in other domains of semantic reanalysis like folk etymologies and puns.

Acknowledgments

We would like to thank the anonymous reviewers for their excellent and insightful comments.

References

Alexander Budanitsky and Graeme Hirst. 2001. Semantic distance in wordnet:an experimental, application-oriented evaluation of five measures. In *Proceedings of the ACL Workshop on WordNet and Other Lexical Resources*.

Christiane Fellbaum, editor. 1998. *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, MA.

Mark Liberman. 2003. Egg corns: folk etymology, malapropism, mondegreen, ??? *http://158.130.17.5/ myl/languagelog/archives/000019.html*.

Ruli Manurung, Graeme Ritchie, Helen Pain, Annalu Waller, Dave O'Mara, and Rolf Black. 2008. The construction of a pun generator for language skills development. *Applied Artificial Intelligence*, 22:841–869.

Rani Nelken and Elif Yamangil. 2008. Mining Wikipedia's article revision history for training computational linguistics algorithms. In *Proceedings of the AAAI Workshop on Wikipedia and Artificial Intelligence*.

Gabriella Rundblad and David B Kronenfeld. 1998. Folk-etymology: Haphazard perversion or shrewd analogy? In Julie Coleman and Christian Kay, editors, *Lexicology, Semantics, and Lexicography*. John Benjamins, Manchester.

Gerard Salton and Christopher Buckley. 1988. Term weighting approaches in automatic text retrieval. *Information Processing and Management*, 24(5):513–523.

Phil Scholfield. 1988. Documenting folk etymological change in progress. *English Studies*, 69:341–347.

Oliviero Stock and Carlo Strapparava. 2006. Laughing with hahacronym, a computational humor system. In *Proceedings of the 21st AAAI Conference on Artificial Intelligence*.